


```
1 0001 0 XTITLE 'LIBSCLI_CALLBACK - CLI Callback Interface Procedures'  
2 0002 0 MODULE LIBSCLI_CALLBACK {  
3 0003 0 IDENT = 'V04-000'      ! CLI Callback Procedures  
4 0004 0 ) =  
5 0005 1 BEGIN  
6 0006 1 *****  
7 0007 1 *  
8 0008 1 *  
9 0009 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY  
10 0010 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.  
11 0011 1 * ALL RIGHTS RESERVED.  
12 0012 1 *  
13 0013 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  
14 0014 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE  
15 0015 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER  
16 0016 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  
17 0017 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY  
18 0018 1 * TRANSFERRED.  
19 0019 1 *  
20 0020 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE  
21 0021 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT  
22 0022 1 * CORPORATION.  
23 0023 1 *  
24 0024 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS  
25 0025 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.  
26 0026 1 *  
27 0027 1 *  
28 0028 1 *****  
29 0029 1  
30 0030 1  
31 0031 1 ++  
32 0032 1 FACILITY: General Utility Library  
33 0033 1  
34 0034 1 ABSTRACT:  
35 0035 1  
36 0036 1 This module contains callable procedures which allow user programs  
37 0037 1 to access the CLI callback facility. The procedures in this module  
38 0038 1 are:  
39 0039 1 LIB$GET_SYMBOL      get value of a CLI symbol  
40 0040 1 LIB$SET_SYMBOL      set value of a CLI symbol  
41 0041 1 LIB$DELETE_SYMBOL   delete a CLI symbol  
42 0042 1 LIB$SET_LOGICAL    set value of a supervisor mode  
43 0043 1                   logical name  
44 0044 1 LIB$DELETE_LOGICAL delete a supervisor mode logical name  
45 0045 1 LIB$DISABLE_CTRL  disable CLI out-of-band handling  
46 0046 1 LIB$ENABLE_CTRL    re-enable CLI out-of-band handling  
47 0047 1  
48 0048 1 ENVIRONMENT: Runs only in USER mode - AST reentrant  
49 0049 1  
50 0050 1 AUTHOR: Ralph O. Weber,      CREATION DATE: 19-AUG-1981  
51 0051 1  
52 0052 1 MODIFIED BY:  
53 0053 1  
54 0054 1 1-001 - Original. ROW 19-AUG-1981  
55 0055 1 1-002 - Change symbol name > 255 characters error to LIB$_INVSYMNAME.  
56 0056 1                   ROW 3-DEC-1981  
57 0057 1 1-003 - Use CLI$C_SRVDESC. Improve code. SBL 18-Dec-1981
```

: 58 0058 1 | 1-004 - Correct all references to LIB\$ANALYZE SDESC. DGP 31-Dec-1981
: 59 0059 1 | 1-005 - Correct PSECT definitions. SBL 4-Jan-1981
: 60 0060 1 | 1-006 - Fix reference to CLIMSG.B32 to use SHRLIB\$, not LIBS. TMH 14-Feb-1982
: 61 0061 1 | 1-007 - Allow symbol name to start with \$ or . SBL 3-Feb-1983
: 62 0062 1 | 1-008 - Use new callbacks to allow itemlists and attributes. STAN 26-Feb-1984.
: 63 0063 1 | 1-009 - Fix bug in way pass itemlist to callback. STAN 10-JUL-1984.
: 64 0064 1 | --

```
66      0065 1 %SBTTL 'Declarations'  
67      0066 1  
68      0067 1      SWITCHES:  
69      0068 1  
70      0069 1  
71      0070 1      SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);  
72      0071 1  
73      0072 1  
74      0073 1      LINKAGES:  
75      0074 1  
76      0075 1  
77      0076 1      LINKAGE LIB$ANALYZE_SDESC JSB LINK = JSB (REGISTER=0; REGISTER=1,REGISTER=2);  
78      0077 1      NOTUSED (3,4,5,6,7,8,9,10,11);  
79      0078 1  
80      0079 1  
81      0080 1      TABLE OF CONTENTS:  
82      0081 1  
83      0082 1  
84      0083 1      FORWARD ROUTINE  
85      0084 1      LIB$GET_SYMBOL,          ! get value of a CLI symbol  
86      0085 1      LIB$SET_SYMBOL,        ! set value of a CLI symbol  
87      0086 1      LIB$SET_LOGICAL,       ! set value of a supervisor-mode  
88      0087 1      logical name  
89      0088 1      LIB$DELETE_LOGICAL,      ! delete a supervisor-mode logical  
90      0089 1      name  
91      0090 1      LIB$DISABLE_CTRL,        ! disable CLI out-of-band recognition  
92      0091 1      LIB$ENABLE_CTRL,         ! re-enable CLI out-of-band recognition  
93      0092 1      LIB$BUILD_SYMBOL_NAME;   ! build a good symbol name  
94      0093 1  
95      0094 1  
96      0095 1      INCLUDE FILES:  
97      0096 1  
98      0097 1  
99      0098 1      LIBRARY 'SY$LIBRARY:STARLET';      ! System symbol definitions  
100     0099 1  
101     0100 1      REQUIRE 'SHRLIBS:CLIMSG';        ! CLIS_ symbols  
102     0380 1  
103     0381 1  
104     0382 1      MACROS:  
105     0383 1  
106     0384 1      MACRO MOVEDESC builds a S-type string descriptor at _TO  
107     0385 1      which points to the same data area as that pointed to by  
108     0386 1      the string descriptor at _FROM. NB: since the _TO  
109     0387 1      descriptor will be used only by SY$CLI which ignores the  
110     0388 1      DTTYPE and CLASS fields, only the LENGTH and POINTER fields  
111     0389 1      are built by MOVEDESC.  
112     0390 1  
113     M 0391 1      MACRO MOVEDESC (_FROM, _TO) =  
114     M 0392 1      BEGIN  
115     M 0393 1      REGISTER  
116     M 0394 1      RET STATUS = 0;  
117     M 0395 1      RET STATUS = LIB$ANALYZE_SDESC R2 ( _FROM;  
118     M 0396 1      BLOCK [_TO, DSCSW_LENGTH:, BYTE]  
119     M 0397 1      BLOCK [_TO, DSCSA_POINTER:, BYTE] );  
120     M 0398 1      IF NOT .RET_STATUS  
121     M 0399 1      THEN  
122     M 0400 1      RETURN (.RET_STATUS);
```

```

123      0401 1      END%;

124      0402 1
125      0403 1
126      0404 1      EQUATED SYMBOLS:
127      0405 1      NONE
128      0406 1
129      0407 1
130      0408 1      FIELDS:
131      0409 1      NONE
132      0410 1
133      0411 1
134      0412 1      PSECTS:
135      0413 1
136      0414 1
137      0415 1      PSECT
138      0416 1      CODE = _LIB$CODE (READ, NOWRITE, EXECUTE, SHARE, PIC, ADDRESSING_MODE (WORD_RELATIVE));
139      0417 1      PLIT = _LIB$CODE (READ, NOWRITE, EXECUTE, SHARE, PIC, ADDRESSING_MODE (WORD_RELATIVE));
140      0418 1      OWN = _LIB$DATA (READ, WRITE, NOEXECUTE, NOSHARE, PIC, ADDRESSING_MODE (LONG_RELATIVE));
141      0419 1      GLOBAL = _LIB$DATA (READ, WRITE, NOEXECUTE, NOSHARE, PIC, ADDRESSING_MODE (LONG_RELATIVE));
142      0420 1
143      0421 1
144      0422 1      OWN STORAGE:
145      0423 1      NONE
146      0424 1
147      0425 1
148      0426 1      EXTERNAL REFERENCES:
149      0427 1
150      0428 1
151      0429 1      EXTERNAL ROUTINE
152      0430 1      SYSSCLI, ! CLI Callback Routine
153      0431 1      LIB$ANALYZE_SDESC_R2 : LIB$ANALYZE_SDESC_JSB LINK,
154      0432 1      LIB$COPY_DXDX, ! Analyze a string descriptor
155      0433 1      LIB$GET1_DD, ! Copy strings of any type
156      0434 1      LIB$FREE1_DD; ! to any type
157      0435 1      LIB$GET1_DD, ! Get a dynamic string
158      0436 1      LIB$FREE1_DD; ! Free a dynamic string
159      0437 1
160      0438 1      EXTERNAL
161      0439 1      LIB$AB_UPCASE; ! the string upcase table
162      0440 1
163      0441 1      EXTERNAL LITERAL
164      0442 1      LIB$INVARG, ! Condition value symbols
165      0443 1      LIB$WRONUMARG, ! Invalid argument
166      0444 1      LIB$INVSYMMNAME, ! Wrong number of arguments
167      0445 1      LIB$NOSUCHSYM, ! Illegal symbol name
168      0446 1      LIB$INSCLIMEM, ! No such symbol
169      0447 1      LIB$AMBSYMDEF, ! Insufficient CLI memory
170      0448 1      LIB$NOCLI, ! Ambiguous symbol defined
171      0449 1      LIB$UNECLIERR; ! No CLI present
172      0450 1      LIB$NOCLI; ! Unexpected CLI error

```

```

173      0450 1 %SBTTL 'LIB$GET_SYMBOL - get value of a CLI symbol'
174      0451 1 GLOBAL ROUTINE [IBSGET_SYMBOL {
175      0452 1     SYMBOL: REF BLOCK [. BYTE],
176      0453 1     RETBUF: REF BLOCK [. BYTE],
177      0454 1     RETLEN: REF VECTOR [, WORD],
178      0455 1     MODE: REF VECTOR [. LONG]
179      0456 1   ) =
180      0457 1
181      0458 1 ++
182      0459 1     FUNCTIONAL DESCRIPTION:
183      0460 1
184      0461 1     LIB$GET_SYMBOL gets the value of the specified CLI symbol and returns
185      0462 1     it in the specified buffer. Before the actual search, however, the
186      0463 1     specified symbol name is upcased so that it will better match symbol
187      0464 1     names formed by the CLI. The local-symbol table is scanned first for
188      0465 1     the symbol, and if no matching symbol is found there, the global-
189      0466 1     symbol table is scanned. The length of the returned value and a mode
190      0467 1     value, indicating the table in which the symbol value was found, are
191      0468 1     optionally returned when parameters to receive them are supplied in
192      0469 1     the calling sequence.
193      0470 1
194      0471 1     Numeric values are automatically translated to decimal strings before
195      0472 1     being returned.
196      0473 1
197      0474 1     The optional mode value can be one of:
198      0475 1     LIB$K_CLI_LOCAL_SYM = 1 ==> Local symbol table name
199      0476 1     LIB$K_CLI_GLOBAL_SYM = 2 ==> Global symbol table name
200      0477 1     These symbols are defined in $LIBCLIDEF.

201      0478 1
202      0479 1     CALLING SEQUENCE:
203      0480 1
204      0481 1     ret_status.wlc.v = LIB$GET_SYMBOL (symbol.rt.dx, retbuf.wt.dx
205      0482 1           [, retlen.ww.r [, mode.wl.r]])
206      0483 1
207      0484 1     FORMAL PARAMETERS:
208      0485 1
209      0486 1     symbol.rt.dx - A string, passed by descriptor, which contains the
210      0487 1     symbol to be searched for. An upcased copy of this
211      0488 1     string will actually be used for the search.
212      0489 1
213      0490 1     retbuf.wt.dx - A string, passed by descriptor, into which the value
214      0491 1     of the symbol, if found, will be written.
215      0492 1
216      0493 1     retlen.ww.r - An optional word which, if present, will receive
217      0494 1     the length of the returned symbol value string.
218      0495 1
219      0496 1     mode.wl.r - An optional longword which, if present, will receive a
220      0497 1     value indicating table in which symbol was found.
221      0498 1     Possible values are:
222      0499 1     LIB$K_CLI_LOCAL_SYM ==> Local symbol table name
223      0500 1     LIB$K_CLI_GLOBAL_SYM ==> Global symbol table name
224      0501 1
225      0502 1     IMPLICIT INPUTS:
226      0503 1
227      0504 1     NONE
228      0505 1
229      0506 1     IMPLICIT OUTPUTS:

```

```

230      0507 1
231      0508 1
232      0509 1
233      0510 1
234      0511 1
235      0512 1
236      0513 1
237      0514 1
238      0515 1
239      0516 1
240      0517 1
241      0518 1
242      0519 1
243      0520 1
244      0521 1
245      0522 1
246      0523 1
247      0524 1
248      0525 1
249      0526 1
250      0527 1
251      0528 1
252      0529 1
253      0530 1
254      0531 2
255      0532 2
256      0533 2
257      0534 2
258      0535 2
259      0536 2
260      0537 2
261      0538 2
262      0539 2
263      0540 2
264      0541 2
265      0542 2
266      0543 2
267      0544 2
268      0545 2
269      0546 2
270      0547 2
271      0548 2
272      0549 2
273      0550 2
274      0551 2
275      0552 2
276      0553 2
277      0554 2
278      0555 2
279      0556 2
280      0557 2
281      0558 2
282      0559 2
283      0560 2
284      0561 2
285      0562 2
286      0563 2

      0507 1
      0508 1
      0509 1
      0510 1
      0511 1
      0512 1
      0513 1
      0514 1
      0515 1
      0516 1
      0517 1
      0518 1
      0519 1
      0520 1
      0521 1
      0522 1
      0523 1
      0524 1
      0525 1
      0526 1
      0527 1
      0528 1
      0529 1
      0530 1
      0531 2
      0532 2
      0533 2
      0534 2
      0535 2
      0536 2
      0537 2
      0538 2
      0539 2
      0540 2
      0541 2
      0542 2
      0543 2
      0544 2
      0545 2
      0546 2
      0547 2
      0548 2
      0549 2
      0550 2
      0551 2
      0552 2
      0553 2
      0554 2
      0555 2
      0556 2
      0557 2
      0558 2
      0559 2
      0560 2
      0561 2
      0562 2
      0563 2

      NONE

      COMPLETION STATUS: (or ROUTINE VALUE:)

      SSS_NORMAL      Normal successful completion
      LIB$_INVARG     Invalid argument
      LIB$_INVSYMNAM  Invalid symbol name
      LIB$_NOSUCHSYM  No such symbol found
      LIB$_INSCLIMEM  Insufficient CLI memory
      LIB$_NOCLI      No CLI present to perform function
      LIB$_UNECLIERR  Unexpected CLI Error

      SIDE EFFECTS:

      WARNING:
      Although this procedure performs some checks on the validity of
      symbol names passed to it, some symbol name considered valid by
      this procedure will be considered invalid by DCL. Callers of this
      procedure are responsible for insuring that symbol names passed to
      this procedure contain only alphanumeric characters.

      --
      BEGIN
      BUILTIN
      NULLPARAMETER;
      LOCAL
      CLI_REQ_BLOCK : BLOCK [CLIS$_SRVDESC, BYTE], ! A CLI request block
      DYN_STRING : BLOCK [8, BYTE], ! Descriptor for dynamic string
      RETURN_STATUS : BLOCK [4, BYTE]; ! A return status value

      !+ Initialize CLI Command request block.
      !-
      CH$FILL (0, CLIS$_SRVDESC, CLI_REQ_BLOCK);
      CLI_REQ_BLOCK [CLIS$B_RQTYPE] = CLIS$K_CLIS$SERV;
      CLI_REQ_BLOCK [CLIS$W_SERVCOD] = CLIS$K_GETSYM;
      RETURN_STATUS = LIB$SBUILD_SYMBOL_NAME( .SYMBOL, DYN_STRING,
      CLI_REQ_BLOCK [CLIS$Q_NAMDESC] );
      IF NOT .RETURN_STATUS THEN RETURN .RETURN_STATUS;

      !+ Get CLI symbol value.
      !-
      RETURN_STATUS = SY$CLI (CLI_REQ_BLOCK);

      !+ Free dynamic string used for upcased symbol name.
      !-
      LIB$FREE1_DD( DYN_STRING );

      !+

```

```

287 0564 2 ! Process returned symbol value and requested optional values.
288 0565 2 !-
289 0566 2 IF .RETURN_STATUS
290 0567 2 THEN
291 0568 2 BEGIN
292 0569 2 LOCAL
293 0570 2     CLI_DESC : REF BLOCK [8, BYTE];
294 0571 2     CLI_DESC = CLI_REQ_BLOCK [CLI$Q_VALDESC];
295 0572 2     RETURN_STATUS = LIB$SCOPY_DDX [CLI_DESC, .RETBUF];
296 0573 2     IF NOT NULLPARAMETER (3) AND .RETURN_STATUS
297 0574 2     THEN
298 0575 4 BEGIN
299 0576 4 LOCAL
300 0577 4     ADDRESS,
301 0578 4     LENGTH;
302 0579 4     LIB$ANALYZE_SDESC_R2 (.RETBUF; LENGTH, ADDRESS );
303 0580 4     RETLEN [0] = MINU (.LENGTH, .CLI_DESC[DSCSW_LENGTH]);
304 0581 3     END;
305 0582 3 IF NOT NULLPARAMETER (4)
306 0583 3 THEN
307 0584 3     MODE [0] = .CLI_REQ_BLOCK [CLISB_RQSTAT];
308 0585 3 END
309 0586 2 ELSE IF (.RETURN_STATUS [STSSV_FAC_NO] EQL CLIS_FACILITY)
310 0587 2 THEN
311 0588 2     RETURN_STATUS =
312 0589 2     BEGIN
313 0590 2         SELECTONE .RETURN_STATUS OF
314 0591 2         SET
315 0592 2             [CLIS_UNDSYM] : LIBS_NOSUCHSYM;
316 0593 2             [CLIS_BUFOVF] : LIBS_INSCLIMEM;
317 0594 2             [CLIS_INVREQTYP] : LIBS_NOCLI;
318 0595 2             [OTHERWISE] : LIBS_UNECLIERR;
319 0596 2             TES
320 0597 2         END;
321 0598 2
322 0599 2     RETURN (.RETURN_STATUS);
323 0600 2
324 0601 1 END;
    
```

! End of routine LIB\$GET_SYMBOL

.TITLE LIB\$CLI_CALLBACK LIB\$CLI_CALLBACK - CLI Callbac
 k Interface Proc

.IDENT \V04-000\

.EXTRN SYSSCLI, LIB\$ANALYZE_SDESC_R2
 .EXTRN LIB\$SCOPY_DDX, LIB\$GET1_DD
 .EXTRN LIB\$SFREET_DD, LIB\$AB_UPCASE
 .EXTRN LIBS_INVARG, LIBS_WRONUMARG
 .EXTRN LIBS_INVSYMMNAM, LIBS_NOSUCHSYM
 .EXTRN LIBS_INSCLIMEM, LIBS_AMBSYMDEF
 .EXTRN LIBS_NOCLI, LIBS_UNECLIERR

.PSECT _LIB\$CODE,NOWRT, SHR, PIC.2

.ENTRY LIB\$GET_SYMBOL, Save R2,R3,R4,R5
 -92(SP)-SP
 MOVAB #0, (SP), #0, #84, CLI_REQ_BLOCK

0054 8F

00

5E A4 003C 00000
 6E 00 9E 00002
 00 2C 00006

: 0451

: 0546

; Routine Size: 213 bytes, Routine Base: _LIB\$CODE + 0000

LIB\$CLI-CALLBACK LIB\$CLI-CALLBACK - CLI Callback Interface Proc 16-Sep-1984 02:22:35
V04-000 LIB\$GET_SYMBOL - get value of a CLI symbol 14-Sep-1984 13:27:47

M 8

VAX-11 Bliss-32 V4.0-742
[VMSLIB.SRC]LIBCLICAL.B32;1

Page (3)

LII
VO

```

326 0602 1 %SBTTL 'LIB$SET_SYMBOL - set value of a CLI symbol'
327 0603 1 GLOBAL ROUTINE [LIB$SET_SYMBOL (
328 0604 1      SYMBOL: REF BLOCK [. BYTE],
329 0605 1      VALUE: REF BLOCK [. BYTE],
330 0606 1      MODE: REF VECTOR [. LONG]
331 0607 1      ) =
332 0608 1      ++
333 0609 1      FUNCTIONAL DESCRIPTION:
334 0610 1
335 0611 1
336 0612 1      This routine defines a CLI symbol giving it the value specified by
337 0613 1      call parameter value. Before the actual definition, however, the
338 0614 1      specified symbol name is upcased so that it will better match symbol
339 0615 1      names formed by the CLI. An attempt to define a CLI symbol which
340 0616 1      already exists results in the value of that symbol being updated to
341 0617 1      the new value. The optional mode argument specifies a code
342 0618 1      (LIB$K_CLI_LOCAL_SYM or LIB$K_CLI_GLOBAL_SYM) which names the symbol
343 0619 1      table on which the define/update operation will be performed. If the
344 0620 1      mode argument is omitted, the local symbol table is used.
345 0621 1
346 0622 1
347 0623 1      The optional mode value can be one of:
348 0624 1      LIB$K_CLI_LOCAL_SYM = 1 ==> Local symbol table name
349 0625 1      LIB$K_CLI_GLOBAL_SYM = 2 ==> Global symbol table name
350 0626 1      These symbols are defined in $LIBCLIDEF.
351 0627 1
352 0628 1      CALLING SEQUENCE:
353 0629 1      ret_status.wlc.v = LIB$SET_SYMBOL (symbol.rt.dx, value.rt.dx
354 0630 1      [, mode.rl.r])
355 0631 1
356 0632 1      FORMAL PARAMETERS:
357 0633 1
358 0634 1      symbol.rt.dx - A string, passed by descriptor, which contains the
359 0635 1      symbol to be defined or modified. An upcased copy of
360 0636 1      this string will actually be used for the define
361 0637 1      or update operation.
362 0638 1
363 0639 1      value.rt.dx - A string, passed by descriptor, containing the value
364 0640 1      to be given to the symbol.
365 0641 1
366 0642 1      mode.rl.r - An optional longword which, if present, contains a
367 0643 1      value indicating into which table the symbol is to be
368 0644 1      placed. If this parameter is omitted, the local
369 0645 1      symbol table is used.
370 0646 1      Possible values are:
371 0647 1      LIB$K_CLI_LOCAL_SYM ==> Local symbol table name
372 0648 1      LIB$K_CLI_GLOBAL_SYM ==> Global symbol table name
373 0649 1
374 0650 1      IMPLICIT INPUTS:
375 0651 1      NONE
376 0652 1
377 0653 1
378 0654 1      IMPLICIT OUTPUTS:
379 0655 1      NONE
380 0656 1
381 0657 1
382 0658 1      COMPLETION STATUS: (or ROUTINE VALUE:)
```

383 0659 1 :
384 0660 1 :
385 0661 1 : SSS_NORMAL Normal successful completion
386 0662 1 : LIBS_INVARG Invalid argument
387 0663 1 : LIBS_INVSYMNAM Invalid symbol name
388 0664 1 : LIBS_INSCLIMEM Insufficient CLI memory
389 0665 1 : LIBS_AMBSYMDFT Ambiguous CLI symbol defined
390 0666 1 : LIBS_NOCLI No CLI present to perform function
391 0667 1 : LIBS_UNECLIERR Unexpected CLI Error

392 0668 1 : SIDE EFFECTS:
393 0669 1 :
394 0670 1 : A CLI symbol is created or updated.
395 0671 1 :
396 0672 1 :
397 0673 1 :
398 0674 1 :
399 0675 1 :
400 0676 1 :
401 0677 1 :
402 0678 1 :
403 0679 1 :
404 0680 1 :
405 0681 1 :
406 0682 1 :--
407 0683 1 :
408 0684 2 : BEGIN
409 0685 2 :
410 0686 2 : BUILTIN
411 0687 2 : NULLPARAMETER:
412 0688 2 :
413 0689 2 :
414 0690 2 : LOCAL CLI_REQ_BLOCK : BLOCK [CLISG_SRVDESC, BYTE], ! A CLI request block
415 0691 2 : DYN_STRING : BLOCK [8, BYTE], ! Descriptor for dynamic string
416 0692 2 : RETURN_STATUS : BLOCK [4, BYTE]; ! A return status value
417 0693 2 :
418 0694 2 :
419 0695 2 :
420 0696 2 :
421 0697 2 :
422 0698 2 :
423 0699 2 : CHSFILL (0, CLISG_SRVDESC, CLI_REQ_BLOCK);
424 0700 2 : CLI_REQ_BLOCK [CLISB_RQTYPE] = CLISK_CLISERV;
425 0701 3 : BEGIN
426 0702 3 : LOCAL
427 0703 3 : STRING : REF BLOCK [8, BYTE];
428 0704 3 : STRING = .VALUE;
429 0705 3 : IF .STRING [DSC\$W_LENGTH] GTRU 255 THEN RETURN LIBS_INVARG;
430 0706 2 : END;
431 0707 2 : MOVEDESC (.VALUE, CLI_REQ_BLOCK [CLISQ_VALDESC]);
432 0708 2 : RETURN_STATUS = LIBSSBUILD_SYMBOL_NAME (.SYMBOL, DYN_STRING,
433 0709 2 : CLI_REQ_BLOCK [CLISQ_NAMDESC]);
434 0710 2 : IF NOT .RETURN_STATUS THEN RETURN .RETURN_STATUS;
435 0711 2 :
436 0712 2 :
437 0713 2 :
438 0714 2 : Setup service code indicating which symbol table.
439 0715 2 :
439 0715 3 :
439 0715 3 : CLI_REQ_BLOCK [CLISW_SERVCOD] =
439 0715 3 : BEGIN

```

: 440      0716 3      IF NULLPARAMETER (3)
: 441      0717 3      THEN
: 442      0718 3      ELSE CLISK_DEFLOCAL
: 443      0719 3
: 444      0720 3      CASE .MODE [0] FROM LIBSK_CLI_LOCAL_SYM
: 445      0721 3          TO LIBSK_CLI_GLOBAL_SYM OF
: 446      0722 3          SET
: 447      0723 3          [LIBSK_CLI_LOCAL_SYM] : CLISK_DEFLOCAL;
: 448      0724 3          [LIBSK_CLI_GLOBAL_SYM] : CLISK_DEFGLOBAL;
: 449      0725 3          [OUTRANGE] : RETURN (LIBS_INVARG);
: 450      0726 3
: 451      0727 3      TES
: 452      0728 3      END;
: 453      0729 2
: 454      0730 2      |+ Set CLI symbol.
: 455      0731 2      |- RETURN_STATUS = SYSSCLI (CLI_REQ_BLOCK);
: 456      0732 2
: 457      0733 2
: 458      0734 2
: 459      0735 2      |+ Free dynamic string used for upcased symbol name.
: 460      0736 2      |- LIBSSFREE1_DD( DYN_STRING );
: 461      0737 2
: 462      0738 2
: 463      0739 2
: 464      0740 2      |+ Adjust error return status, if any.
: 465      0741 2      |- IF NOT .RETURN_STATUS AND (.RETURN_STATUS [STSSV_FAC_NO] EQ CLIS_FACILITY)
: 466      0742 2
: 467      0743 2      THEN
: 468      0744 2          RETURN_STATUS =
: 469      0745 2          BEGIN
: 470      0746 2          |+ SELECTONE .RETURN_STATUS OF
: 471      0747 2          |+ SET
: 472      0748 2          [CLIS_SYMOVF] : LIBS_INSCLIMEM;
: 473      0749 2          [CLIS_ABSYMD] : LIBS_AMBSYMD;
: 474      0750 2          [CLIS_INVREQTYP] : LIBS_NOCLI;
: 475      0751 2          [OTHERWISE] : LIBS_UNECLIERR;
: 476      0752 2
: 477      0753 2      TES
: 478      0754 2      END;
: 479      0755 2      RETURN (.RETURN_STATUS);
: 480      0756 2
: 481      0757 1      END;

```

! End of routine LIBSSET_SYMBOL

| | | | | | | | | | |
|------|----|------|----|-----------|-------|--------|--------|----------------------------------|------|
| 0054 | 8F | 00 | 5E | A4 | 003C | 00000 | .ENTRY | LIBSSET_SYMBOL, Save R2,R3,R4,R5 | 0603 |
| | | | 6E | 00 | 9E | 00002 | MOVAB | -92(SP), SP | 0698 |
| | | | | 08 | 2C | 00006 | MOVCS | #0, (SP), #0, #84, CLI_REQ_BLOCK | |
| | | | | AE | 0000D | | | | |
| | | | 08 | AE | 0000F | | | | |
| | | | 50 | 08 | 90 | 000013 | MOVBL | #5, CLI_REQ_BLOCK | 0699 |
| | | 00FF | 8F | AC | 00 | 00017 | CMPW | VALUE, STRING | 0703 |
| | | | | 60 | B1 | 00017 | BGTRU | (STRING), #255 | 0704 |
| | | | 50 | 3D | 1A | 0001C | MOVL | 3S | |
| | | | | 08 | 00 | 0001E | JSB | VALUE, R0 | 0706 |
| | | | | 00000000G | 16 | 00022 | | LIBSANALYZE_SDESC_R2 | |

| | | | | | | | |
|-----------|----------|------|-----------|-------|--------|--------------------------------|------|
| 14 | AE | 51 | B0 | 00028 | MOVW | R1, CLI_REQ_BLOCK+12 | |
| 18 | AE | 52 | DD | 0002C | MOVL | R2, CLI_REQ_BLOCK+16 | |
| | 01 | 50 | E8 | 00030 | BLBS | REF_STATUS, 1\$ | |
| | | | 04 | 00033 | RET | | |
| | | 0C | AE | 9F | PUSHAB | CLI_REQ_BLOCK+4 | 0708 |
| | | 04 | AE | 9F | PUSHAB | DYN_STRING | 0707 |
| | | 04 | AC | DD | PUSHL | SYMBOL | 0708 |
| 00000V | CF | 03 | FB | 0003D | CALLS | #3, LIB\$BUILD_SYMBOL_NAME | |
| | 52 | 50 | DD | 00042 | MOVL | RO, RETURN_STATUS | |
| | 7B | 52 | E9 | 00045 | BLBC | RETURN_STATUS, 9\$ | 0709 |
| | 03 | 6C | 91 | 00048 | CMPB | (AP), #3 | 0716 |
| | | 16 | 1F | 0004B | BLSSU | 4\$ | |
| | | 0C | AC | D5 | TSTL | 12(AP) | |
| 01 | 01 | 11 | 13 | 00050 | BEQL | 4\$ | |
| | 0011 | 0C | BC | CF | CASEL | AMODE, #1, #1 | 0720 |
| | | 000C | 00052 | 00057 | .WORD | 4\$-2\$, - | |
| | | 50 | 00000000G | 8F | MOVL | #LIBS_INVARG, RO | 0725 |
| | | | | 04 | RET | | |
| | | 50 | 02 | DD | MOVL | #2, RO | 0720 |
| | | | 03 | 11 | BRB | 6\$ | |
| 09 | AE | 50 | 03 | DD | MOVL | #3, RO | |
| | 08 | 50 | B0 | 0006B | MOVW | RO, CLI_REQ_BLOCK+1 | 0715 |
| 00000000G | 00 | 01 | FB | 00072 | PUSHAB | CLI_REQ_BLOCK | 0732 |
| | 52 | 50 | DD | 00079 | CALLS | #1, SYSSCLI | |
| 00000000G | 00 | 5E | DD | 0007C | MOVL | RO, RETURN_STATUS | |
| | 44 | 01 | FB | 0007E | PUSHL | SP | 0737 |
| 03 | 52 | 0C | 52 | E8 | CALLS | #1, LIB\$FREE1_DD | |
| | 00038138 | 10 | ED | 00088 | BLBS | RETURN_STATUS, 11\$ | 0742 |
| | | 3D | 12 | 0008D | CMPZV | #16, #T2, RETURN_STATUS, #3 | |
| | | 52 | D1 | 0008F | BNEQ | 11\$ | |
| | | 09 | 12 | 00096 | CMPL | RETURN_STATUS, #229688 | 0748 |
| | | 52 | DD | 00098 | MOVL | 7\$ | |
| | 000381A0 | 8F | 2B | 11 | BRB | #LIBS_INSCLIMEM, RETURN_STATUS | |
| | | 52 | D1 | 000A1 | 7S: | 11\$ | |
| | | 09 | 12 | 000A8 | CMPL | RETURN_STATUS, #229792 | 0749 |
| | | 52 | DD | 000AA | BNEQ | 8\$ | |
| 00038822 | 8F | 19 | 11 | 000B1 | MOVL | #LIBS_AMBSYMDEF, RETURN_STATUS | |
| | | 52 | D1 | 000B3 | 8S: | 11\$ | |
| | | 09 | 12 | 000BA | BRB | RETURN_STATUS, #231458 | 0750 |
| | | 52 | DD | 000BC | CMPL | 10\$ | |
| | | 07 | 11 | 000C3 | BNEQ | #LIBS_NOCLI, RETURN_STATUS | |
| | | 52 | DD | 000C5 | 9S: | 11\$ | |
| | | 04 | 000CF | 10\$: | MOVL | #LIBS_UNECLIERR, RETURN_STATUS | 0751 |
| | | 50 | DD | 000CC | 11\$: | RET | 0755 |
| | | | | | MOVL | RETURN_STATUS, RO | 0757 |

; Routine Size: 208 bytes, Routine Base: _LIB\$CODE + 0005

```

483 0758 1 %SBTTL 'LIB$DELETE_SYMBOL - delete a CLI symbol'
484 0759 1 GLOBAL ROUTINE LIB$DELETE_SYMBOL (
485 0760 1   SYMBOL: REF BLOCK[, BYTE],           ! set value of a CLI symbol
486 0761 1   MODE: REF VECTOR [, LONG]          ! symbol string descriptor
487 0762 1 ) =                                ! desired symbol table (opt.)

```

488 0763 1
489 0764 1
490 0765 1
491 0766 1
492 0767 1
493 0768 1
494 0769 1
495 0770 1
496 0771 1
497 0772 1
498 0773 1
499 0774 1
500 0775 1
501 0776 1
502 0777 1
503 0778 1
504 0779 1
505 0780 1
506 0781 1
507 0782 1
508 0783 1
509 0784 1
510 0785 1
511 0786 1
512 0787 1
513 0788 1
514 0789 1
515 0790 1
516 0791 1
517 0792 1
518 0793 1
519 0794 1
520 0795 1
521 0796 1
522 0797 1
523 0798 1
524 0799 1
525 0800 1
526 0801 1
527 0802 1
528 0803 1
529 0804 1
530 0805 1
531 0806 1
532 0807 1
533 0808 1
534 0809 1
535 0810 1
536 0811 1
537 0812 1
538 0813 1
539 0814 1

++ FUNCTIONAL DESCRIPTION:

This routine deletes a CLI symbol. Before the actual deletion, however, the specified symbol name is upcased so that it will better match symbol names formed by the CLI. The optional mode argument specifies a code (LIBSK_CLI_LOCAL_SYM or LIBSK_CLI_GLOBAL_SYM) which names the symbol table on which the delete operation will be performed. If the mode argument is omitted, the local symbol table is used.

The optional mode value can be one of:

LIBSK_CLI_LOCAL_SYM = 1 ==> Local symbol table name
LIBSK_CLI_GLOBAL_SYM = 2 ==> Global symbol table name

These symbols are defined in \$LIBCLIDEF.

CALLING SEQUENCE:

ret_status.wlc.v = LIBSDELETE_SYMBOL (symbol.rt.dx [, mode.rl.r])

FORMAL PARAMETERS:

symbol.rt.dx - A string, passed by descriptor, which contains the symbol to be defined or modified. An upcased copy of this string will actually be used for the deletion.

mode.rl.r - An optional longword which, if present, contains a value indicating from which table the symbol is to be deleted. If this parameter is omitted, the local symbol table is used.

Possible values are:

LIBSK_CLI_LOCAL_SYM ==> Local symbol table name
LIBSK_CLI_GLOBAL_SYM ==> Global symbol table name

IMPLICIT INPUTS:

NONE

IMPLICIT OUTPUTS:

NONE

COMPLETION STATUS: (or ROUTINE VALUE:)

| | |
|------------------|------------------------------------|
| SS\$ NORMAL | Normal successful completion |
| LIB\$ INVARG | Invalid argument |
| LIB\$_INVSYNMNAM | Invalid symbol name |
| LIB\$_NOSUCHSYM | No such symbol found |
| LIB\$_NOCLI | No CLI present to perform function |
| LIB\$_UNECLIERR | Unexpected CLI Error |

SIDE EFFECTS:

```
540      0815 1 | A CLI symbol is deleted.
541      0816 1 |
542      0817 1 |
543      0818 1 |
544      0819 1 |
545      0820 1 |
546      0821 1 |
547      0822 1 |
548      0823 1 |
549      0824 1 |
550      0825 1 |-
551      0826 1 |
552      0827 2 |-
553      0828 2 |
554      0829 2 |
555      0830 2 |
556      0831 2 |
557      0832 2 |
558      0833 2 | CLI_REQ_BLOCK : BLOCK [CLIS$C_SRVDESC, BYTE], ! A CLI request block
559      0834 2 | DYN$STRING : BLOCK [8, BYTE] ! Descriptor for dynamic string
560      0835 2 | RETURN_STATUS : BLOCK [4, BYTE]; ! A return status value
561      0836 2 |
562      0837 2 |
563      0838 2 |
564      0839 2 |+
565      0840 2 | Initialize CLI Command request block.
566      0841 2 |-
567      0842 2 | CHSFILL (0, CLIS$C_SRVDESC, CLI_REQ_BLOCK);
568      0843 2 | RETURN_STATUS = LIB$SBUILD_SYMBOL_NAME(.SYMBOL, DYN$STRING,
569      0844 2 |                               CLI_REQ_BLOCK[CLIS$Q_NAMDESC]);
570      0845 2 | IF NOT .RETURN_STATUS THEN RETURN .RETURN_STATUS;
571      0846 2 | CLI_REQ_BLOCK[CLIS$B_RQTYPE] = CLIS$K_CLISERV;
572      0847 2 |
573      0848 2 |+
574      0849 2 | Setup service code indicating which symbol table.
575      0850 2 |-
576      0851 2 | BEGIN
577      0852 2 | IF NULLPARAMETER (2)
578      0853 2 | THEN
579      0854 2 | ELSE CLIS$K_DELELCL
580      0855 2 | CASE .MODE [0] FROM LIBSK$CLI_LOCAL_SYM
581      0856 2 |                      TO LIBSK$CLI_GLOBAL_SYM OF
582      0857 2 | SET
583      0858 2 | [LIBSK$CLI_LOCAL_SYM] : CLIS$K_DELELCL;
584      0859 2 | [LIBSK$CLI_GLOBAL_SYM] : CLIS$K_DELEGBL;
585      0860 2 | [OUTRANGE] : RETURN (LIB$INVARG);
586      0861 2 |
587      0862 2 | TES
588      0863 2 | END;
589      0864 2 |
590      0865 2 |+
591      0866 2 | Delete CLI symbol.
592      0867 2 |-
593      0868 2 | RETURN_STATUS = SYS$CLI (CLI_REQ_BLOCK);
594      0869 2 |
595      0870 2 |+
596      0871 2 | Free dynamic string used for upcased symbol name.
```

```

597 0872 2 !-
598 0873 2 LIB$FREE1_DD( DYN_STRING );
599 0874 2
600 0875 2
601 0876 2
602 0877 2
603 0878 2
604 0879 2
605 0880 2
606 0881 2
607 0882 2
608 0883 2
609 0884 2
610 0885 2
611 0886 2
612 0887 2
613 0888 2
614 0889 2
615 0890 2
616 0891 2
617 0892 1

    !- Adjust error return status, if any.
    IF NOT .RETURN_STATUS AND (.RETURN_STATUS [STSSV_FAC_NO] EQL CLIS_FACILITY)
    THEN
        RETURN_STATUS =
        BEGIN
            SELECTONE .RETURN_STATUS OF
            SET
            [CLIS_UNDSYM] : LIBS_NOSUCHSYM;
            [CLIS_INVREQTYP] : LIBS_NOCLI;
            [OTHERWISE] : LIBS_UNECLIERR;
            TES
        END;
    RETURN (.RETURN_STATUS);
    END;

```

! End of routine LIBSDELETE_SYMBOL

| | | | | | | | | |
|------|----|-----------|------|------|---------------|-----------------------------------|--|------|
| 0054 | BF | 00 | 5E | A4 | 003C | 00000 | .ENTRY LIBSDELETE_SYMBOL, Save R2,R3,R4,R5 | 0759 |
| | | | | 08 | AE | 9E 00002 | MOVAB -92(SP) SP | 0841 |
| | | | | 08 | AE | 00 2C 00006 | MOVCS #0, (SP), #0, #84, CLI_REQ_BLOCK | 0841 |
| | | | | 0C | AE | 9F 0000F | PUSHAB CLI_REQ_B'OCK+4 | 0843 |
| | | | | 04 | AE | 9F 00012 | PUSHAB DYN_STRING | 0842 |
| | | | | 04 | AC | DD 00015 | PUSHL SYMBOL | 0843 |
| | | 0000V | CF | 03 | FB | 00018 | CALLS #3, LIB\$BUILD_SYMBOL_NAME | |
| | | | 52 | 50 | 00 | 0001D | MOVL R0, RETURN_STATUS | |
| | | | 76 | 52 | E9 | 00020 | BLBC RETURN_STATUS, 7\$ | |
| | | 08 | AE | 05 | 90 | 00023 | MOVB #5, CLI_REQ_BLOCK | 0844 |
| | | | 02 | 6C | 91 | 00027 | (MPB (AP), #2 | 0845 |
| | | | | 16 | 1F | 0002A | BLSSU 2\$ | 0852 |
| | | | | 08 | AC | D5 0002C | TSTL 8(AP) | |
| | | | | 11 | 13 | 0002F | BEQL 2\$ | |
| | | 01 | 01 | 08 | BC | CF 00031 | CASEL @MODE, #1, #1 | 0856 |
| | | | 0011 | 000C | | 00036 1\$: | .WORD 2\$-1\$, - | |
| | | | | 50 | 00000000G | 8F D0 0003A | .WORD 3\$-1\$ | |
| | | | | | | 04 00041 | MOVL #LIBS_INVARG, R0 | 0861 |
| | | | | 50 | | 08 00 00042 2\$: | RET | |
| | | | | | | 03 11 00045 | MOVL #11, R0 | 0856 |
| | | 09 | 50 | 0C | 00 | 00047 3\$: | BRB 4\$ | |
| | | | AE | 50 | B0 0004A 4\$: | MOVL #12, R0 | | |
| | | | 08 | AE | 9F 0004F | MOVW R0, CLI_REQ_BLOCK+1 | 0851 | |
| | | 00000000G | 00 | 01 | FB 00051 | PUSHAB CLI_REQ_BLOCK | 0868 | |
| | | | 52 | 50 | DD 00058 | CALLS #1, SYSSCLI | | |
| | | 00000000G | 00 | 5E | DD 00058 | MOVL R0, RETURN_STATUS | | |
| | | | 32 | 01 | FB 0005D | PUSHL SP | 0873 | |
| | | | | 52 | E8 00064 | CALLS #1, LIB\$FREE1_DD | | |
| 03 | 52 | 0C | | 10 | ED 00067 | BLBS RETURN_STATUS, 7\$ | | |
| | | | | | | CMPZV #16, #T2, RETURN_STATUS, #3 | 0878 | |

LIBSCLI_CALLBACK LIBSCLI_CALLBACK - CLI Callback Interface Proce 16-Sep-1984 02:22:35
V04-000 LIBSDELETE_SYMBOL - delete a CLI symbol 14-Sep-1984 13:27:47 H 9 VAX-11 Bliss-32 V4.0-742
[VMSLIB.SRC]LIBCLICAL.B32;1

Page 17
(5)

| | | | | | | |
|----------|--------------|-------------|------|--------------------------------|--------------------------------|------|
| 00038140 | 8F | 28 12 0006C | BNEQ | 78 | RETURN_STATUS, #229696 | 0884 |
| | | 52 D1 0006E | CMPL | | | |
| | | 09 12 00075 | BNEQ | 58 | | |
| | 52 00000000G | 8F D0 00077 | MOVL | #LIBS_NOSUCHSYM, RETURN_STATUS | | |
| | | 19 11 0007E | BRB | 78 | | |
| 00038822 | 8F | 52 D1 00080 | 58: | CMPL | RETURN_STATUS, #231458 | 0885 |
| | | 09 12 00087 | BNEQ | 68 | | |
| | 52 00000000G | 8F D0 00089 | MOVL | #LIBS_NOCLI, RETURN_STATUS | | |
| | | 07 11 00090 | BRB | 78 | | |
| | 52 00000000G | 8F D0 00092 | 68: | MOVL | #LIBS_UNECLIERR, RETURN_STATUS | 0886 |
| | | 52 D0 00099 | 78: | MOVL | RETURN_STATUS, #0 | 0890 |
| | | 04 0009C | RET | | | 0892 |

: Routine Size: 157 bytes, Routine Base: _LIB\$CODE + 01A5

```

: 619 0893 1 %SBTTL 'LIB$SET_LOGICAL - set supervisor mode logical name'
: 620 0894 1 GLOBAL ROUTINE [LIB$SET_LOGICAL (
: 621 0895 1      LOGNAME : REF $BBLOCK,           ! set supervisor mode logical name
: 622 0896 1      VALUE   : REF $BBLOCK,           ! logical name string descriptor
: 623 0897 1      TABNAM  : REF $BBLOCK,           ! equivalence name string descriptor
: 624 0898 1      ATTR    : REF $BBLOCK[4],        ! logical name table
: 625 0899 1      ITMLST  : REF $BBLOCK           ! attributes
: 626 0900 1           ) =                      ! Address of itemlist
: 627
: 628
: 629
: 630
: 631
: 632
: 633
: 634
: 635
: 636
: 637
: 638
: 639
: 640
: 641
: 642
: 643
: 644
: 645
: 646
: 647
: 648
: 649
: 650
: 651
: 652
: 653
: 654
: 655
: 656
: 657
: 658
: 659
: 660
: 661
: 662
: 663
: 664
: 665
: 666
: 667
: 668
: 669
: 670
: 671
: 672
: 673
: 674
: 675

```

0901 1 ++ FUNCTIONAL DESCRIPTION:

0902 1
0903 1 This routine provides a method by which a non-privileged program can
0904 1 create/modify supervisor mode logical names. The logical name given
0905 1 by logname is created or modified to have the equivalence name
0906 1 specified by value.
0907 1 The user may specify attributes for the logical name and may
0908 1 specify which logical name table is to be used.
0909 1 Instead of specifying a single equivalence string, via VALUE,
0910 1 the caller may specify an item list to give multiple equivalence strings.

0911 1
0912 1 The STRNLNM system service can be used to obtain the value of
0913 1 supervisor mode logical names. The routine is equivalent to the ASSIGN
0914 1 or DEFINE DCL command.

0915 1 CALLING SEQUENCE:

```

0916 1
0917 1
0918 1
0919 1
0920 1
0921 1
0922 1
0923 1
0924 1
0925 1
0926 1
0927 1
0928 1
0929 1
0930 1
0931 1
0932 1
0933 1
0934 1
0935 1
0936 1
0937 1
0938 1
0939 1
0940 1
0941 1
0942 1
0943 1
0944 1
0945 1
0946 1
0947 1
0948 1
0949 1

```

ret_status.wlc.v = LIB\$SET_LOGICAL (

- logname.rt.dx
- [. value.rt.dx]
- [. tabnam.rt.dx]
- [. attr.rlu.r]
- [. itmlst.ra.r])

0928 1 FORMAL PARAMETERS:

0929 1 logname.rt.dx - A string, passed by descriptor, which contains the
0930 1 supervisor mode logical name to be created or modified.

0932 1 value.rt.dx - A string, passed by descriptor, specifying the value
0933 1 to be given to the supervisor mode logical name.
0934 1 If omitted, an itemlist must be present to specify the
0935 1 value(s) of the logical name.

0937 1 tabnam.rt.dx - Name of the table in which to create the logical name.
0938 1 If omitted, the LNMSPROCESS table is used.

0939 1 attr.rlu.r - Logical name attributes. See the description of
0940 1 the system service \$CRELNM for more details.
0941 1 Currently available attributes are
0942 1 LNMSM_CONFINE and LNMSM_NO_ALIAS.
0943 1 If omitted, no special attributes are established.
0944 1 However, if no table or itemlist is specified,
0945 1 then the attribute LNMSM_CRELOG is defaulted in.

0946 1 If no itemlist is specified, then the translation
0947 1 attributes LNMSM_CONCEALED and LNMSM_TERMINAL may

676 0950 1 also be specified here. They apply to the
 677 0951 1 equivalence string string specified by the VALUE
 678 0952 1 parameter.
 679 0953 1
 680 0954 1 If an itemlist is specified, then the itemlist will
 681 0955 1 contain the translation attributes for each
 682 0956 1 equivalence string in the itemlist.
 683 0957 1
 684 0958 1 itemlist.ra.r The address of an item list describing the
 685 0959 1 equivalence name(s) for this logical name.
 686 0960 1 See the description of the \$CRELNRM system service
 687 0961 1 for the format and meaning of the item list.
 688 0962 1
 689 0963 1 If omitted, the logical name will have just one
 690 0964 1 value, as specified by the VALUE parameter.
 691 0965 1
 692 0966 1 Either VALUE or ITMLST must be specified.
 693 0967 1 If neither are specified, the LIBS_INVARG
 694 0968 1 error is produced.
 695 0969 1
 696 0970 1 If both are specified, then the VALUE parameter
 697 0971 1 is ignored. In this case, logical name
 698 0972 1 attributes are permitted, but translation
 699 0973 1 attributes specified in the ATTR parameter
 700 0974 1 will be ignored (since these will appear in the
 701 0975 1 itemlist).
 702 0976 1
 703 0977 1 The ITMLST parameter is only needed in the case where
 704 0978 1 you wish to create multiple equivalence strings
 705 0979 1 for a single logical name (i.e. a search list).
 706 0980 1
 707 0981 1 IMPLICIT INPUTS:
 708 0982 1 It is assumed that the logical name table specified already exists.
 709 0983 1
 710 0984 1 IMPLICIT OUTPUTS:
 711 0985 1
 712 0986 1
 713 0987 1
 714 0988 1
 715 0989 1
 716 0990 1 COMPLETION STATUS: (or ROUTINE VALUE:
 717 0991 1 SSS_NORMAL Normal successful completion
 718 0992 1 SSS_SUPERSEDE Previous logical name replaced (success)
 719 0993 1 SSS_ACCVIO logname or value cannot be read
 720 0994 1 SSS_IVLOGNAM logname or value contains more than 255 characters
 721 0995 1 SSS_BADPARAM One or more arguments had a bad value.
 722 0996 1 For example, an unknown bit was specified in
 723 0997 1 the attributes parameter.
 724 0998 1 Note that if an item list is specified,
 725 0999 1 then translation attributes (if any) must
 726 1000 1 appear in the item list and may not appear
 727 1001 1 in the ATTR parameter. Only logical name
 728 1002 1 attributes may appear in the ATTR parameter
 729 1003 1 in that case.
 730 1004 1 The specified logical name table does not exist.
 731 1005 1 The caller lacks the necessary privileges
 732 1006 1 to create the logical name.

```

733 1007 1 | SSS_TOOMANYLNAM Logical name translation of the table name exceeded
734 1008 1 | the allowable depth (10 translations).
735 1009 1 | SSS_DUPLNAM Attempt to create a logical name that conflicts
736 1010 1 | with a logical name created at a higher access mode
737 1011 1 | and which had the LNMSM_NO_ALIAS attribute.
738 1012 1 | SSS_EXLNMQOUTA Exceeded logical name space quota.
739 1013 1 | SSS_INSMEM Insufficient dynamic memory.
740 1014 1 | LIB$_INVARG Invalid argument. Neither a VALUE nor an ITMLST
741 1015 1 | was specified.
742 1016 1 | Or LOGNAME was omitted.
743 1017 1 | No CLI present to perform function
744 1018 1 | Unexpected CLI Error
745 1019 1 | LIBS_WRONUMARG Wrong number of arguments. Either fewer than two
746 1020 1 | arguments or more than 5 arguments were specified.
747 1021 1 |
748 1022 1 | SIDE EFFECTS:
749 1023 1 | A supervisor mode logical name is created or modified.
750 1024 1 |
751 1025 1 | --
752 1026 1 |
753 1027 1 |
754 1028 2 | BEGIN
755 1029 2 |
756 1030 2 | BUILTIN
757 1031 2 |
758 1032 2 | ACTUALCOUNT,
759 1033 2 | NULLPARAMETER;
760 1034 2 |
761 1035 2 | LOCAL
762 1036 2 |
763 1037 2 | CLI_REQ_BLOCK : BLOCK [CLISC_SRVDESC, BYTE], ! A CLI request block
764 1038 2 | RETURN_STATUS : BLOCK [4, BYTE],
765 1039 2 | TRANSLATION_ATTRIBUTES,
766 1040 2 | LOGICAL_NAME_ATTRIBUTES,
767 1041 2 | SPECIAL_ITEM_LIST : VECTOR [?]; ! Special item list
768 1042 2 |
769 1043 2 |+
770 1044 2 | Check for proper number of arguments.
771 1045 2 |-
772 1046 2 |
773 1047 2 | IF ACTUALCOUNT() LEQU 1
774 1048 2 | OR ACTUALCOUNT() GTRU 5
775 1049 2 | THEN
776 1050 2 | RETURN LIBS_WRONUMARG;
777 1051 2 |
778 1052 2 |+
779 1053 2 | Initialize CLI command request block accounting for the differences
780 1054 2 | between creating/modifying a logical name and deleting one.
781 1055 2 |
782 1056 2 |
783 1057 2 | CHSFILL (0, CLISC_SRVDESC, CLI_REQ_BLOCK);
784 1058 2 | CLI_REQ_BLOCK [CLISB_RQTYPE] = CLISK_CLISERV;
785 1059 2 | CLI_REQ_BLOCK [CLISW_SERVCOD] = CLISR_CREALOG;
786 1060 2 |
787 1061 2 |+
788 1062 2 | Move in the logical name, the equivalence string, and the logical
789 1063 2 | name table name.

```

```

790 1064 2 | Give an error if the logical name is omitted.
791 1065 2 | Give an error if neither VALUE nor ITMLST is supplied.
792 1066 2
793 1067 2
794 1068 2
795 1069 2
796 1070 2
797 1071 2
798 1072 2
799 1073 2
800 1074 2
801 1075 2
802 1076 2
803 1077 2
804 1078 2
805 1079 2
806 1080 2
807 1081 2
808 1082 2
809 1083 2
810 1084 2
811 1085 2
812 1086 2
813 1087 2
814 1088 2
815 1089 2
816 1090 2
817 1091 2
818 1092 2
819 1093 2
820 1094 2
821 1095 2
822 1096 2
823 1097 2
824 1098 2
825 1099 2
826 1100 2
827 1101 2
828 1102 2
829 1103 2
830 1104 2
831 1105 2
832 1106 2
833 1107 2
834 1108 2
835 1109 2
836 1110 2
837 1111 2
838 1112 2
839 1113 2
840 1114 2
841 1115 2
842 1116 2
843 1117 2
844 1118 2
845 1119 2
846 1120 2

    IF .LOGNAME EQL 0
    THEN
        RETURN LIBS_INVARG;

    MOVEDESC ( .LOGNAME, CLI_REQ_BLOCK [CLISQ_NAMDESC] );

    IF NULLPARAMETER(VALUE) AND NULLPARAMETER(ITMLST)
    THEN
        RETURN LIBS_INVARG;

    IF NOT NULLPARAMETER(VALUE)
    THEN
        MOVEDESC ( .VALUE, CLI_REQ_BLOCK [CLISQ_VALDESC] );

    +
    | NOTE:
    | If both VALDESC and ITMLST are specified, DCL will ignore VALUE
    |

    +
    | Move in the table name if present.
    | If not present, DCL defaults to LNMSPROCESS.
    |

    IF NOT NULLPARAMETER(TABNAM)
    THEN
        MOVEDESC ( .TABNAM, CLI_REQ_BLOCK [CLISQ_TABDESC] );

    +
    | Move in the address of the item list (if present).
    | If not present, DCL will create an itemlist from CLISQ_VALDESC.
    |

    IF NOT NULLPARAMETER(ITMLST)
    THEN
        CLI_REQ_BLOCK [CLISL_ITMLST] = .ITMLST;

    +
    | Move in the address of the attributes (if present).
    |

    IF NOT NULLPARAMETER(ATTR)
    THEN
        CLI_REQ_BLOCK [CLISL_ATTR] = .ATTR;

    +
    | NOTE:
    | If CLISQ_TABDESC, CLISL_ITMLST, and CLISL_ATTR are all 0,
    | then DCL defaults in the attribute LNMSM_CRELOG, i.e.
    | the call back looks like a V3 call back.
    |

```

```

847 1121 2 | If the caller has specified translation attributes in ATTR
848 1122 2 | (as opposed to just logical name attributes), this is normally
849 1123 2 | an error. That is because translation attributes should go in
850 1124 2 | the itemlist. However, as a convenience to RTL users who find
851 1125 2 | it hard to build itemlists, we allow translation attributes to
852 1126 2 | occur if no itemlist is specified. In that case, we have to go
853 1127 2 | through the trouble of building an itemlist for him.
854 1128 2 |
855 1129 2 |
856 1130 2 | IF NULLPARAMETER(ITMLST) AND NOT NULLPARAMETER(ATTR)
857 1131 2 | THEN
858 1132 2 | BEGIN ! Check for translation attributes
859 1133 2 | BIND ATTR_VECTOR = .ATTR : VECTOR[4,BYTE],
860 1134 2 | ATTR_VALUE = .ATTR : LONG;
861 1135 2 |
862 1136 2 |
863 1137 2 | The first byte of attributes are logical name attributes.
864 1138 2 | The second byte of attributes are translation attributes.
865 1139 2 | The other 2 bytes are reserved for VMS use.
866 1140 2 |
867 1141 2 |
868 1142 2 | IF .ATTR_VECTOR[1] NEQ 0
869 1143 2 | THEN
870 1144 4 | BEGIN ! Build special itemlist
871 1145 4 | BIND EQUIV_STRING_DESC = CLI_REQ_BLOCK[CLISQ_VALDESC] : SBBLOCK;
872 1146 4 |
873 1147 4 |
874 1148 4 | ! Create translation attributes from the specified attributes.
875 1149 4 |
876 1150 4 |
877 1151 4 | TRANSLATION_ATTRIBUTES=.ATTR_VALUE AND XX'FF00';
878 1152 4 |
879 1153 4 |
880 1154 4 | ! Create logical name attributes from the specified attributes.
881 1155 4 |
882 1156 4 |
883 1157 4 | LOGICAL_NAME_ATTRIBUTES=.ATTR_VALUE AND XX'FF';
884 1158 4 |
885 1159 4 | SPECIAL_ITEM_LIST[0]=LNMS_ATTRIBUTES^16+4;
886 1160 4 | SPECIAL_ITEM_LIST[1]=TRANSLATION_ATTRIBUTES;
887 1161 4 | SPECIAL_ITEM_LIST[2]=0;
888 1162 4 | SPECIAL_ITEM_LIST[3]=LNMS_STRING^16+.EQUIV_STRING_DESC[DSCSW_LENGTH];
889 1163 4 | SPECIAL_ITEM_LIST[4]=EQUIV_STRING_DESC[DSCSA_POINTER];
890 1164 4 | SPECIAL_ITEM_LIST[5]=0;
891 1165 4 | SPECIAL_ITEM_LIST[6]=0; ! no more entries
892 1166 4 |
893 1167 4 |
894 1168 4 | ! Store specially revised attributes and item list in
895 1169 4 | the request block.
896 1170 4 |
897 1171 4 |
898 1172 4 | CLI_REQ_BLOCK[CLISL_ATTR]=LOGICAL_NAME_ATTRIBUTES;
899 1173 4 | CLI_REQ_BLOCK[CLISL_ITMLST]=SPECIAL_ITEM_LIST
900 1174 4 |
901 1175 4 | END ! Build special itemlist
902 1176 4 |
903 1177 2 | END; ! Check for translation attributes

```

```

904 1178 2
905 1179 2
906 1180 2 Create or modify the supervisor mode logical name.
907 1181 2
908 1182 2 RETURN_STATUS = SYSSCLI (CLI_REQ_BLOCK);
909 1183 2
910 1184 2
911 1185 2 Adjust error return status, if any.
912 1186 2
913 1187 2 IF NOT .RETURN_STATUS AND (.RETURN_STATUS [STSSV_FAC_NO] EQL CLIS_FACILITY)
914 1188 2 THEN
915 1189 2 RETURN_STATUS =
916 1190 2 BEGIN
917 1191 2     SELECTONE .RETURN_STATUS OF
918 1192 2     SET
919 1193 2     [CLIS_INVREQTYP] : LIBS_NOCLI;
920 1194 2     [OTHERWISE] : LIBS_UNECLI;ERR;
921 1195 2     TES
922 1196 2     END;
923 1197 2
924 1198 2 RETURN .RETURN_STATUS
925 1199 2
926 1200 1 END: ! End of LIBSSET_LOGICAL
    
```

| | | | | | ENTRY | LIBSSET_LOGICAL, Save R2,R3,R4,R5,R6 | | 0894 |
|------|----|----|--------------|------------------|-------|--------------------------------------|--|------|
| 0054 | 8F | 00 | 56 00000000G | 00 9E 00002 | MOVAB | LIB_NA[YZE SDESC_R2, R6 | | |
| | | | 5E 88 | AE 9E 00009 | MOVAB | -12U(SP), SP | | |
| | | | 01 | 6C 91 0000D | CMPB | (AP), #1 | | 1047 |
| | | | 05 | 05 1B 00010 | BLEQU | 1\$ | | |
| | | | 50 00000000G | 08 1B 00015 | CMPB | (AP), #5 | | 1048 |
| | | | 8F | D0 00017 1\$: | BLEQU | 2\$ | | |
| | | | | 04 0001E | MOVL | #LIBS_WRONUMARG, R0 | | 1050 |
| | | | | | RET | | | |
| | | | 6E | 00 2C 0001F 2\$: | MOVCS | #0, (SP), #0, #84, CLI_REQ_BLOCK | | 1057 |
| | | | 24 | AE 00026 | MOVB | #5, CLI_REQ_BLOCK | | |
| | | | 25 | AE 05 90 00028 | MOVW | #6, CLI_REQ_BLOCK+1 | | 1058 |
| | | | | 06 B0 0002C | TSTL | LOGNAME | | 1059 |
| | | | 04 | AC D5 00030 | BEQL | 4\$ | | 1068 |
| | | | 50 | 25 13 00033 | MOVL | LOGNAME, R0 | | |
| | | | | AC D0 00035 | JSB | LIBSANALYZE_SDESC R2 | | 1072 |
| | | | 66 | 66 16 00039 | MOVW | R1, CLI_REQ_BLOCK+4 | | |
| | | | 51 | B0 0003B | MOVL | R2, CLI_REQ_BLOCK+8 | | |
| | | | 2C | 52 D0 0003F | BLBC | REF STATUS, 7\$ | | |
| | | | 4F | 50 E9 00043 | CMPB | (AP), #2 | | 1074 |
| | | | 02 | 6C 91 00046 | BLSSU | 3\$ | | |
| | | | | 05 1F 00049 | TSTL | 8(AP) | | |
| | | | 08 | AC D5 0004B | BNEQ | 5\$ | | |
| | | | | 12 12 0004F | BLSSU | 4\$ | | |
| | | | 05 | 6C 91 00050 3\$: | CMPB | (AP), #5 | | |
| | | | | 05 1F 00053 | TSTL | 20(AP) | | |
| | | | 14 | AC D5 00055 | BNEQ | 5\$ | | |
| | | | | 08 12 00058 | MOVL | #LIBS_INVARG, R0 | | 1076 |
| | | | 50 00000000G | 8F D0 0005A 4\$: | | | | |

| | | | | | | |
|----|----|-------------|----------|---|---|------|
| 02 | 08 | 06 00061 | RET | | 1078 | |
| | | 6C 91 00062 | CMPB | (AP), #2 | | |
| | | 16 1F 00065 | BLSSU | 6S | | |
| | 08 | D5 00067 | TSTL | 8(AP) | | |
| 50 | 08 | 11 13 0006A | BEQL | 6S | | |
| | | AC 00 0006C | MOVL | VALUE, R0 | 1080 | |
| 30 | AE | 66 16 00070 | JSB | LIBSANALYZE_SDESC R2 | | |
| 34 | AE | 51 B0 00072 | MOVW | R1, CLI_REQ_BLOCK+12 | | |
| 18 | | 52 D0 00076 | MOVL | R2, CLI_REQ_BLOCK+16 | | |
| 03 | | 50 E9 0007A | BLBC | RET_STATUS, 7S | | |
| | | 6C 91 0007D | CMPB | (AP), #3 | 1092 | |
| | 0C | 17 1F 00080 | BLSSU | 8S | | |
| | 0C | AC D5 00082 | TSTL | 12(AP) | | |
| 50 | 0C | 12 13 00085 | BEQL | 8S | | |
| | | AC 00 00087 | MOVL | TABNAM, R0 | 1094 | |
| 38 | AE | 66 16 0008B | JSB | LIBSANALYZE_SDESC R2 | | |
| 3C | AE | 51 B0 0008D | MOVW | R1, CLI_REQ_BLOCK+20 | | |
| 01 | | 52 D0 00091 | MOVL | R2, CLI_REQ_BLOCK+24 | | |
| | | 50 E8 00095 | BLBS | RET_STATUS, 8S | | |
| | 05 | 04 00098 | RET | | | |
| | | 6C 91 00099 | CMPB | (AP), #5 | 1101 | |
| | | 0A 1F 0009C | BLSSU | 9S | | |
| | 14 | AC D5 0009E | TSTL | 20(AP) | | |
| 40 | AE | 05 13 000A1 | BEQL | 9S | | |
| 04 | | AC 00 000A3 | MOVL | ITMLST, CLI_REQ_BLOCK+28 | | |
| | | 6C 91 000A8 | CMPB | (AP), #4 | 1103 | |
| | | 0A 1F 000AB | BLSSU | 10S | 1109 | |
| | 10 | AC D5 000AD | TSTL | 16(AP) | | |
| 44 | AE | 05 13 000B0 | BEQL | 10S | | |
| 05 | | AC 00 000B2 | MOVL | ATTR, CLI_REQ_BLOCK+32 | | |
| | | 6C 91 000B7 | CMPB | (AP), #5 | 1111 | |
| | 10 | 05 1F 000BA | BLSSU | 11S | 1130 | |
| | | AC D5 000BC | TSTL | 20(AP) | | |
| | 14 | 4B 12 000BF | BNEQ | 12S | | |
| | 04 | 6C 91 000C1 | CMPB | (AP), #4 | | |
| | | 46 1F 000C4 | BLSSU | 12S | | |
| | | AC D5 000C6 | TSTL | 16(AP) | | |
| | 10 | 41 13 000C9 | BEQL | 12S | | |
| | 50 | AC 00 000CB | MOVL | ATTR, R0 | | |
| | 01 | A0 95 000CF | TSTB | 1(R0) | 1133 | |
| | | 38 13 000D2 | BEQL | 12S | 1142 | |
| 6E | 10 | BC FFFF00FF | BICL3 | #65281, ATTR, TRANSLATION ATTRIBUTES | 1151 | |
| 04 | AE | 10 BC | MOVZBL | ATTR, LOGICAL NAME ATTRIBUTES | 1157 | |
| 08 | AE | 00030004 | 8F D0 | #196612, SPECIAL ITEM LIST | 1159 | |
| 0C | AE | | 000E2 | TRANSLATION ATTRIBUTES, SPECIAL_ITEM_LIST+4 | 1160 | |
| 14 | AE | 10 AE | 6E 9E | CLRL | 1161 | |
| 16 | AE | 30 AE | D4 000EE | SPECIAL_ITEM_LIST+8 | 1162 | |
| 18 | AE | 34 AE | 3C 000F1 | MOVZWL | | |
| 44 | AE | 04 AE | 02 A0 | EQUIV STRING_DESC, SPECIAL_ITEM_LIST+12 | | |
| 40 | AE | 08 AE | 000F6 | ADDW2 | #2, SPECIAL_ITEM_LIST+12 | |
| | | 24 AE | 7C 000FF | MOVBL | EQUIV STRING_DESC+4, SPECIAL_ITEM_LIST+16 | |
| | | 00 0010C | 00102 | CLRL | SPECIAL_ITEM_LIST+20 | |
| | | 01 FB | 0010F | MOVAB | LOGICAL_NAME_ATTRIBUTES, CLI_REQ_BLOCK+32 | |
| | | 50 E8 | 00116 | MOVAB | SPECIAL_ITEM_LIST, CLI_REQ_BLOCK+28 | |
| | | 10 ED | 00119 | PUSHAB | CLI_REQ_BLOCK | |
| | | 18 12 | 0011E | CALLS | #1, SYSCLI | |
| | | | | BLBS | RETURN_STATUS, 14S | |
| 03 | 50 | | | CMPZV | #16, #T2, RETURN_STATUS, #3 | 1187 |
| | | | | BNEQ | 14S | |

LIB\$CLI_CALLBACK LIB\$CLI_CALLBACK - CLI callback Interface Proce 16-Sep-1984 02:22:35
V04-000 LIB\$SET_LOGICAL - set supervisor mode logical n 14-Sep-1984 13:27:47 C 10
[VMSLIB.SRC]LIBCLICAL.B32;1

Page 25
(6)

| | | | | | | | |
|----------|-----------|----|----|------------|------|--------------------------------|--------|
| 00038822 | 8F | 50 | D1 | 00120 | CMPL | RETURN_STATUS, #231458 | : 1193 |
| | | 08 | 12 | 00127 | BNEQ | 138 | |
| 50 | 00000000G | 8F | 00 | 00129 | MOVL | #LIBS_NOCLI, RETURN_STATUS | |
| | | | 04 | 00130 | RET | | |
| 50 | 00000000G | 8F | 00 | 00131 138: | MOVL | #LIBS_UNECLIERR, RETURN_STATUS | : 1194 |
| | | | 04 | 00138 148: | RET | | : 1200 |

: Routine Size: 313 bytes. Routine Base: _LIB\$CODE + 0242

```

928      1201 1 %SBTTL 'LIB$DELETE_LOGICAL - delete supervisor mode logical name'
929      1202 1 GLOBAL ROUTINE LIB$DELETE_LOGICAL (
930          LOGNAME : REF $BBLOCK,
931          TABNAM : REF $BBLOCK
932          ) =
933
934      1207 1 ++
935      1208 1 | FUNCTIONAL DESCRIPTION:
936      1209 1
937      1210 1 | This routine provides a method by which a non-privileged program can
938      1211 1 | delete a supervisor mode logical name. The logical name given by
939      1212 1 | logname is deleted from the specified or implied logical name table.
940      1213 1
941      1214 1 | The STRNLOG system service can be used to obtain the value of
942      1215 1 | supervisor mode logical names. The routine is equivalent to the
943      1216 1 | DEASSIGN DCL command.
944      1217 1
945      1218 1 | CALLING SEQUENCE:
946      1219 1
947      1220 1 |     ret_status.wlc.v = LIB$DELETE_LOGICAL (logname.rt.dx [,tabnam.rt.dx])
948      1221 1
949      1222 1 | FORMAL PARAMETERS:
950      1223 1
951      1224 1 | logname.rt.dx - A string, passed by descriptor, which contains the
952      1225 1 | supervisor mode logical name to be deleted.
953      1226 1 | Must be present.
954      1227 1
955      1228 1 | tabnam.rt.dx - A string, passed by descriptor, that contains the
956      1229 1 | name of the logical name table from which the
957      1230 1 | logical name is to be deleted.
958      1231 1 | If omitted, the logical name is removed from
959      1232 1 | the LNMSPROCESS logical name table.
960      1233 1
961      1234 1 | IMPLICIT INPUTS:
962      1235 1
963      1236 1 | It is assumed that the logical name table specified already exists.
964      1237 1
965      1238 1 | IMPLICIT OUTPUTS:
966      1239 1
967      1240 1 |     NONE
968      1241 1
969      1242 1 | COMPLETION STATUS: (or ROUTINE VALUE:)
970      1243 1
971      1244 1 |     SSS_NORMAL      Normal successful completion
972      1245 1 |     SSS_ACCVIO      logname cannot be read
973      1246 1 |     SSS_IVLOGNAM    logname contains more than 63 characters
974      1247 1 |     SSS_NOLOGNAM    No such logical name defined or logical name specified
975      1248 1 |                 was declared in executative or kernel mode
976      1249 1 |     SSS_IVLOGTAB    Invalid logical name table specified
977      1250 1 |     SSS_NOPRIV      Caller lacks the necessary privilege to delete
978      1251 1 |                 the logical name
979      1252 1 |     SSS_TOOMANYLNAM Logical name translation of the table name exceeded
980      1253 1 |                 the allowable depth (10 translations)
981      1254 1 |     LIBS_INVARG     Invalid argument
982      1255 1 |     LIBS_NOCLI      No CLI present to perform function
983      1256 1 |     LIBS_UNECLIERR   Unexpected CLI Error
984      1257 1 |     LIBS_WRONUMARG   No arguments specified or more than two specified.

```

```

985      1258 1 | SIDE EFFECTS:
986      1259 1 | A supervisor mode logical name is deleted from the specified or
987      1260 1 | implied logical name table.
988      1261 1 |
989      1262 1 |
990      1263 1 |
991      1264 1 |
992      1265 1 |
993      1266 2 |-- BEGIN
994      1267 2 |
995      1268 2 | BUILTIN
996      1269 2 |   ACTUALCOUNT,
997      1270 2 |   NULLPARAMETER;
998      1271 2 |
999      1272 2 |
1000     1273 2 | LOCAL
1001     1274 2 |   CLI_REQ_BLOCK : BLOCK [CLISC_SRVDESC, BYTE], ! A CLI request block
1002     1275 2 |   RETURN_STATUS : BLOCK [4, BYTE];
1003     1276 2 |
1004     1277 2 |+ Check for proper number of arguments.
1005     1278 2 |-
1006     1279 2 |
1007     1280 2 | IF ACTUALCOUNT() EQLU 0
1008     1281 2 | OR ACTUALCOUNT() GTRU 2
1009     1282 2 | THEN
1010     1283 2 |   RETURN LIBS_WRONUMARG;
1011     1284 2 |
1012     1285 2 |+ Initialize CLI Command request block.
1013     1286 2 |-
1014     1287 2 |   CHSFILL (0, CLISC_SRVDESC, CLI_REQ_BLOCK);
1015     1288 2 |   CLI_REQ_BLOCK [CLISB_RQTYPE] = CLISK_CLISERV;
1016     1289 2 |   CLI_REQ_BLOCK [CLISU_SERVCOD] = CLISR_DELELOG;
1017     1290 2 |
1018     1291 2 |+ Move in the logical name.
1019     1292 2 |-
1020     1293 2 |
1021     1294 2 | MOVEDESC ( .LOGNAME, CLI_REQ_BLOCK [CLISQ_NAMDESC] );
1022     1295 2 |
1023     1296 2 |
1024     1297 2 |+ Move in the table name, if specified.
1025     1298 2 | If not specified, DCL will default in LNMSPROCESS.
1026     1299 2 |-
1027     1300 2 |
1028     1301 2 |
1029     1302 2 | IF NOT NULLPARAMETER(TABNAM)
1030     1303 2 | THEN
1031     1304 2 |   MOVEDESC ( .TABNAM, CLI_REQ_BLOCK [CLISQ_TABDESC] );
1032     1305 2 |
1033     1306 2 |
1034     1307 2 |+ Delete the supervisor mode logical name.
1035     1308 2 |-
1036     1309 2 |   RETURN_STATUS = SYSSCLI (CLI_REQ_BLOCK);
1037     1310 2 |
1038     1311 2 |+ Adjust error return status, if any.
1039     1312 2 |-
1040     1313 2 |
1041     1314 2 |

```

1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327

```
THEN
  RETURN_STATUS =
BEGIN
  SELECTONE .RETURN_STATUS OF
  SET
  [CLIS INVREQTYP] : LIBS_NOCLI:
  [OTHERWISE] : LIBS_UNECLIERR:
  TES
END;

RETURN .RETURN_STATUS
END;
```

! END OF LIBSDELETE LOGICAL

0054 8F

00

| | | | 007C | 00000 |
|------|-----------|----|------|-------|
| 56 | 00000000G | 00 | 9E | 00002 |
| 5E | AC | AE | 9E | 00009 |
| | | 6C | 95 | 0000D |
| | | 05 | 13 | 0000F |
| 02 | | 6C | 91 | 00011 |
| | | 08 | 1B | 00014 |
| 50 | 00000000G | 8F | D0 | 00016 |
| | | | 04 | 0001D |
| 6E | | 00 | 2C | 0001E |
| | | 6E | | 00025 |
| 01 | 6E | 05 | 90 | 00026 |
| | AE | 07 | B0 | 00029 |
| | 50 | 04 | D0 | 0002D |
| | | 66 | 16 | 00031 |
| 04 | AE | 51 | B0 | 00033 |
| 08 | AE | 52 | D0 | 00037 |
| | 46 | 50 | E9 | 0003B |
| | 02 | 6C | 91 | 0003E |
| | | 16 | 1F | 00041 |
| | | 08 | AC | 00043 |
| | 50 | 08 | 11 | 00046 |
| | | | AC | 00048 |
| 14 | AE | 66 | 16 | 0004C |
| 18 | AE | 51 | B0 | 0004E |
| | 28 | 52 | D0 | 00052 |
| 000G | 00 | 50 | E9 | 00056 |
| | 1F | 5E | DD | 00059 |
| | 0C | | | 38: |
| 822 | 8F | 01 | FB | 0005B |
| | | 50 | E8 | 00062 |
| | | 1C | ED | 00065 |
| | | 18 | 12 | 0006A |
| | | 50 | D1 | 0006C |
| | | 08 | 12 | 00073 |
| 50 | 00000000G | 8F | D0 | 00075 |
| | | | 04 | 0007C |
| 50 | 00000000G | 8F | D0 | 0007D |
| | | | 04 | 00084 |
| | | | 04 | 00088 |

```

.ENTRY LIBSDELETE_LOGICAL, Save R2,R3,R4,R5,R6
MOVAB LIBSANALYZE_SDESC_R2, R6
MOVAB -84(SP), SP
TSTB (AP)
BEQL 1$ 
(CMPB, #2
BLEQU 2$ 
MOVL #LIBS_WRONUMARG, R0
RET
MOVC5 #0, (SP), #0, #84, CLI_REQ_BLOCK

MOVB #5, CLI_REQ_BLOCK
MOVW #7, CLI_REQ_BLOCK+1
MOVL LOGNAME, R0
JSB LIBSANALYZE_SDESC R2
MOVW R1, CLI_REQ_BLOCK+4
MOVL R2, CLI_REQ_BLOCK+8
BLBC REF_STATUS, 5$ 
(CMPB, #2
BLSSU 3$ 
TSTL 8(AP)
BEQL 3$ 
MOVL TABNAM, R0
JSB LIBSANALYZE_SDESC R2
MOVW R1, CLI_REQ_BLOCK+20
MOVL R2, CLI_REQ_BLOCK+24
BLBC REF_STATUS, 5$ 
PUSHL SP
CALLS #1, SYSSCLI
BLBS RETURN_STATUS, 5$ 
(CMPZV #16, #T2, RETURN_STATUS, #3
BNEQ 5$ 
CMPL RETURN_STATUS, #231458
BNEQ 4$ 
MOVL #LIBS_NOCLI, RETURN_STATUS
RET
MOVL #LIBS_UNECLIERR, RETURN_STATUS
BET

```

LIB\$CLI_CALLBACK LIB\$CLI_CALLBACK - CLI Callback Interface Proc 16-Sep-1984 02:22:35
V04-000 LIB\$DELETE_LOGICAL - delete supervisor mode log 14-Sep-1984 13:27:47

G 10

VAX-11 Bliss-32 V4.0-742
[VMSLIB.SRC]LIBCLICAL.B32:1

Page 29
(7)

: Routine Size: 133 bytes, Routine Base: _LIB\$CODE + 0378

1056 1328 1 XSBTTL 'LIB\$DISABLE_CTRL - disable CLI out-of-band character(s) handler'
1057 1329 1 GLOBAL ROUTINE LIB\$DISABLE_CTRL () disable CLI out-of-band
1058 1330 1 DISABLE_MASK : REF VECTOR [, LONG] disable bit mask
1059 1331 1 PREVIOUS_MASK : REF VECTOR [, LONG] previous enable bit mask
1060 1332 1) =

1061 1333 1 ++ FUNCTIONAL DESCRIPTION:

1062 1334 1 This routine disables CLI processing for out-of-band the characters
1063 1335 1 identified by DISABLE_MASK. When normal CLI processing for one or
1064 1336 1 more out-of-band characters is disabled, a user program can intercept
1065 1337 1 and respond to them. There are no privilege requirements for calling
1066 1338 1 this routine.

1067 1339 1 Currently, this routine supports disabling of CLI out-of-band
1068 1340 1 processing for CTRL-T and CTRL-Y when the DCL CLI is in use, and for
1069 1341 1 CTRL-Y when the MCR CLI is in use. The macro \$LIBCLIDEF defines the
1070 1342 1 two masks used for disabling processing for these two out-of-band
1071 1343 1 characters as follows:

1072 1344 1 LIBSM_CLI_CTRLT mask for CTRL-T
1073 1345 1 LIBSM_CLI_CTRLY mask for CTRL-Y
1074 1346 1 If the masks are logically or'd together, processing for both
1075 1347 1 out-of-band characters will be effected.

1076 1348 1 CALLING SEQUENCE:

1077 1349 1 ret_status.wlc.v = LIB\$DISABLE_CTRL (disable_mask.rl.r
1078 1350 1 [, previous_mask.wl.r])

1079 1351 1 FORMAL PARAMETERS:

1080 1352 1 disable_mask.rl.r - the mask controlling which out-of-band
1081 1353 1 characters the CLI should ignore. Values
1082 1354 1 should be selected from one of, or the logical
1083 1355 1 OR of more than one of:
1084 1356 1 LIBSM_CLI_CTRLT
1085 1357 1 LIBSM_CLI_CTRLY

1086 1358 1 These symbols are defined in \$LIBCLIDEF.

1087 1359 1 previous_mask.wl.r - a longword into which will be stored a mask
1088 1360 1 having the same format as disable_mask, but
1089 1361 1 which represents the enabled state for all
1090 1362 1 out-of-band character processing before
1091 1363 1 disable_mask is applied.

1092 1364 1 IMPLICIT INPUTS:

1093 1365 1 NONE

1094 1366 1 IMPLICIT OUTPUTS:

1095 1367 1 NONE

1096 1368 1 COMPLETION STATUS: (or ROUTINE VALUE:)

1097 1369 1 SSS_NORMAL Normal successful completion

1113 1385 1 | LIBS_INVARG Disable_mask has an unrecognized bit set
1114 1386 1 | LIBS_NOCLI No CLI present to perform function
1115 1387 1 | LIBS_UNECLERR Unexpected CLI Error
1116 1388 1
1117 1389 1 | SIDE EFFECTS:
1118 1390 1 | CLI out-of-band processing of one or more characters is disabled.
1119 1391 1
1120 1392 1 |--
1121 1393 1
1122 1394 1
1123 1395 2 | BEGIN
1124 1396 2
1125 1397 2
1126 1398 2 | BUILTIN
1127 1399 2 | NULLPARAMETER;
1128 1400 2
1129 1401 2 | LOCAL
1130 1402 2 | CLI_REQ_BLOCK : BLOCK [CLISC_SRVDESC, BYTE], ! A CLI request block
1131 1403 2 | RETURN_STATUS : BLOCK [4, BYTE];
1132 1404 2
1133 1405 2
1134 1406 2 |+ Initialize CLI Command request block.
1135 1407 2 |
1136 1408 2 | CH\$FILL {0, CLISC_SRVDESC, CLI_REQ_BLOCK};
1137 1409 2 | CLI_REQ_BLOCK [CLISB_RQTYPE] = CLISK_CLISERV;
1138 1410 2 | CLI_REQ_BLOCK [CLISW_SERVOD] = CLISR_DISAOOB;
1139 1411 2 | CLI_REQ_BLOCK [CLISL_NEW_MASK] = .DISABLE_MASK [0];
1140 1412 2
1141 1413 2
1142 1414 2 |+ Request CLI out-of-band processing be disabled.
1143 1415 2 |
1144 1416 2 | RETURN_STATUS = SYSSCLI (CLI_REQ_BLOCK);
1145 1417 2
1146 1418 2
1147 1419 2 |+ If requested, return previous out-of-band enabled mask.
1148 1420 2 | It may be good.
1149 1421 2 |
1150 1422 2 | IF NOT NULLPARAMETER (2)
1151 1423 2 | THEN
1152 1424 2 | PREVIOUS_MASK [0] = .CLI_REQ_BLOCK [CLISL_OLD_MASK];
1153 1425 2
1154 1426 2
1155 1427 2 |+ Adjust error return status, if any.
1156 1428 2 |
1157 1429 2 | IF NOT .RETURN_STATUS AND (.RETURN_STATUS [STSSV_FAC_NO] EQL CLIS_FACILITY)
1158 1430 2 | THEN
1159 1431 2 | RETURN_STATUS =
1160 1432 2 | BEGIN
1161 1433 2 | | SELECTONE .RETURN_STATUS OF
1162 1434 2 | | SET
1163 1435 2 | | [CLIS_BADCTLMSK] : LIBS_INVARG;
1164 1436 2 | | [CLIS_INVREQTYP] : LIBS_NOCLI;
1165 1437 2 | | [OTHERWISE] : LIBS_UNECLERR;
1166 1438 2 | | TES
1167 1439 2 | | END;
1168 1440 2
1169 1441 2 | RETURN (.RETURN_STATUS);

1170 1442 ?
1171 1443 1 END;

! End of LIBSDISABLE_CTRL

| | | | | | | | | | |
|------|----|-----------|--------------|----|------|----------|--------|-------------------------------------|------|
| 0054 | 8F | 00 | 5E | AC | 003C | 00000 | .ENTRY | LIB\$DISABLE_CTRL, Save R2,R3,R4,R5 | 1329 |
| | | | 6E | | AE | 9E 00002 | MOVAB | -84(SP) SP | 1408 |
| | | | | | 00 | 2C 00006 | MOVCS | #0, (SP), #0, #84, CLI_REQ_BLOCK | |
| | | | | | 6F | 0000D | | | |
| | | 01 | 6E | | 05 | 90 0000E | MOVBL | #5, CLI_REQ_BLOCK | 1409 |
| | | 04 | AE | 04 | 0D | B0 00011 | MOVW | #13, CLI_REQ_BLOCK+1 | 1410 |
| | | | AE | | BC | D0 00015 | MOVL | ADISABLE_MASK, CLI_REQ_BLOCK+4 | 1411 |
| | | | | | 5E | DD 0001A | PUSHL | SP | 1416 |
| | | 00000000G | 00 | | 01 | FB 0001C | CALLS | #1, SYSSCLI | |
| | | | 02 | | 6C | 91 00023 | CMPB | (AP), #2 | 1422 |
| | | | | | 0A | 1F 00026 | BLSSU | 1S | |
| | | | | 08 | AC | D5 00028 | TSTL | 8(AP) | |
| | | | | | 05 | 13 0002B | BEQL | 1S | |
| | | 08 | BC | 08 | AE | D0 0002D | MOVL | CLI_REQ_BLOCK+8, @PREVIOUS_MASK | 1424 |
| 03 | | 50 | 30 | | 50 | E8 00032 | BLBS | RETURN_STATUS, 4S | 1429 |
| | | | 0C | | 10 | ED 00035 | CMPZV | #16, #T2, RETURN_STATUS, #3 | |
| | | | | | 29 | 12 0003A | BNEQ | 4S | |
| | | 000388CA | 8F | | 50 | D1 0003C | CMPL | RETURN_STATUS, #231626 | 1435 |
| | | | | | 08 | 12 00043 | BNEQ | 2S | |
| | | | 50 00000000G | 8F | D0 | 00045 | MOVL | #LIBS_INVARG, RETURN_STATUS | |
| | | 00038822 | 8F | | 04 | 0004C | RET | | |
| | | | | | 50 | D1 0004D | CMPL | RETURN_STATUS, #231458 | 1436 |
| | | | | | 08 | 12 00054 | BNEQ | 3S | |
| | | | 50 00000000G | 8F | D0 | 00056 | MOVL | #LIBS_NOCLI, RETURN_STATUS | |
| | | | | | 04 | 0005D | RET | | |
| | | | 50 00000000G | 8F | D0 | 0005E | MOVL | #LIBS_UNECLIERR, RETURN_STATUS | 1437 |
| | | | | | 04 | 00065 | RET | | 1443 |

; Routine Size: 102 bytes, Routine Base: _LIB\$CODE + 0400

```

1173 1444 1 %SBTTL 'LIBSENABLE_CTRL - re-enable CLI out-of-band character(s) handler'
1174 1445 1 GLOBAL ROUTINE LIBSENABLE_CTRL (
1175 1446 1     ENABLE MASK : REF VECTOR [, LONG]      | re-enable CLI out-of-band
1176 1447 1     PREVIOUS_MASK : REF VECTOR [, LONG]  | enable bit mask
1177 1448 1     ) =
1178 1449 1
1179 1450 1 ++
1180 1451 1 | FUNCTIONAL DESCRIPTION:
1181 1452 1
1182 1453 1 | This routine re-enables CLI processing for out-of-band the characters
1183 1454 1 | identified by ENABLE_MASK, presumably after such processing has been
1184 1455 1 | disabled by a call to LIB$DISABLE_CTRL. There are no privilege
1185 1456 1 | requirements for calling this routine.
1186 1457 1
1187 1458 1 | Currently, this routine supports re-enabling of CLI out-of-band
1188 1459 1 | processing for CTRL-T and CTRL-Y when the DCL CLI is in use, and for
1189 1460 1 | CTRL-Y when the MCR CLI is in use. The macro $LIBCLIDEF defines the
1190 1461 1 | two masks used for enabling processing for these two out-of-band
1191 1462 1 | characters as follows:
1192 1463 1 |     LIBSM_CLI_CTRLT      mask for CTRL-T
1193 1464 1 |     LIBSM_CLI_CTRLY      mask for CTRL-Y
1194 1465 1 | If the masks are logically or'd together, processing for both
1195 1466 1 | out-of-band characters will be effected.
1196 1467 1
1197 1468 1 | CALLING SEQUENCE:
1198 1469 1
1199 1470 1 |     ret_status.wlc.v = LIBSENABLE_CTRL (enable_mask.rl.r
1200 1471 1 |                           [, previous_mask.wl.r] )
1201 1472 1
1202 1473 1 | FORMAL PARAMETERS:
1203 1474 1
1204 1475 1 |     enable_mask.rl.r      - the mask controlling which out-of-band
1205 1476 1 |                               characters the CLI should process. Values
1206 1477 1 |                               should be selected from one of, or the logical
1207 1478 1 |                               OR of more than one of:
1208 1479 1 |         LIBSM_CLI_CTRLT
1209 1480 1 |         LIBSM_CLI_CTRLY
1210 1481 1 | These symbols are defined in $LIBCLIDEF.
1211 1482 1
1212 1483 1 |     previous_mask.wl.r      - a longword into which will be stored a mask
1213 1484 1 |                               having the same format as enable_mask, but
1214 1485 1 |                               which represents the enabled state for all
1215 1486 1 |                               out-of-band character processing before
1216 1487 1 |                               enable_mask is applied.
1217 1488 1
1218 1489 1 | IMPLICIT INPUTS:
1219 1490 1 |     NONE
1220 1491 1
1221 1492 1 | IMPLICIT OUTPUTS:
1222 1493 1 |     NONE
1223 1494 1
1224 1495 1
1225 1496 1
1226 1497 1 | COMPLETION STATUS: (or ROUTINE VALUE:)
1227 1498 1
1228 1499 1 |     SSS_NORMAL      Normal successful completion
1229 1500 1 |     LIB$INVARG    Enable_mask has an unrecognized bit set

```

```
1230      1501 1 | LIBS_NOCLI    No CLI present to perform function
1231      1502 1 | LIBS_UNECLERR  Unexpected CLI Error
1232
1233
1234      1504 1 | SIDE EFFECTS:
1235      1505 1 |   CLI out-of-band processing of one or more characters is enabled.
1236      1506 1 |
1237      1507 1 | --
1238      1508 1 |
1239      1509 1 |
1240      1510 2 | BEGIN
1241      1511 2 |
1242      1512 2 | BUILTIN
1243      1513 2 |   NULLPARAMETER;
1244      1514 2 |
1245      1515 2 | LOCAL
1246      1516 2 |   CLI_REQ_BLOCK : BLOCK [CLISC_SRVDESC, BYTE], ! A CLI request block
1247      1517 2 |   RETURN_STATUS : BLOCK [4, BYTE];
1248
1249
1250      1520 2 | Initialize CLI Command request block.
1251      1521 2 |
1252      1522 2 |-
1253      1523 2 |   CHSFILL (0, CLISC_SRVDESC, CLI_REQ_BLOCK);
1254      1524 2 |   CLI_REQ_BLOCK [CLISB_RQTYPE] = CLISK_CLISERV;
1255      1525 2 |   CLI_REQ_BLOCK [CLISW_SERVOD] = CLISR_ENABOOB;
1256      1526 2 |   CLI_REQ_BLOCK [CLISL_NEW_MASK] = .ENABLE_MASK [0];
1257
1258      1528 2 | Request CLI out-of-band processing be enabled.
1259      1529 2 |
1260      1530 2 |-
1261      1531 2 |   RETURN_STATUS = SYSSCLI (CLI_REQ_BLOCK);
1262
1263      1533 2 |+
1264      1534 2 |   If requested, return previous out-of-band enabled mask.
1265      1535 2 |-
1266      1536 2 |   It may be good.
1267      1537 2 |-
1268      1538 2 |   IF NOT NULLPARAMETER (2)
1269      1539 2 |   THEN
1270      1540 2 |     PREVIOUS_MASK [0] = .CLI_REQ_BLOCK [CLISL_OLD_MASK];
1271
1272      1542 2 |+
1273      1543 2 |   Adjust error return status, if any.
1274      1544 2 |-
1275      1545 2 |   IF NOT .RETURN_STATUS AND (.RETURN_STATUS [STSSV_FAC_NO] EQL CLIS_FACILITY)
1276      1546 2 |   THEN
1277      1547 2 |     RETURN_STATUS =
1278      1548 2 |     BEGIN
1279      1549 2 |       SELECTONE .RETURN_STATUS OF
1280      1550 2 |         SET
1281      1551 2 |         [CLIS_BADCTLMSK] : LIBS_INVARG;
1282      1552 2 |         [CLIS_INVREQTYP] : LIBS_NOCLI;
1283      1553 2 |         [OTHERWISE] : LIBS_UNECLERR;
1284      1554 2 |         TES
1285      1555 2 |     END;
1286      1556 2 |
1287      1557 2 |     RETURN (.RETURN_STATUS);
```

LIB\$CLI_CALLBACK - CLI Callback Interface Proc 16-Sep-1984 02:22:35 M 10
 LIB\$ENABLE_CTRL - re-enable CLI out-of-band cha 14-Sep-1984 13:27:47 VAX-11 Bliss-32 V4.0-742
 V04-000 [VMSLIB.SRC]LIBCLICAL.B32:1
 : 1287 1558 1 END: ! End of LIB\$ENABLE_CTRL

Page 35
(9)

| | | | | | | | | |
|------|----|-------------|----|----|-------------|--------|------------------------------------|--------|
| 0054 | 8F | 00 | 5E | AC | 003C 00000 | .ENTRY | LIB\$ENABLE_CTRL, Save R2,R3,R4,R5 | ; 1445 |
| | | | 6E | | AE 9E 00002 | MOVAB | -84(SP) SP | |
| | | | | | 00 2C 00006 | MOVCS | #0, (SP), #0, #84, CLI_REQ_BLOCK | 1523 |
| | | | 01 | AE | 05 90 0000E | MOVB | #5, CLI_REQ_BLOCK | 1524 |
| | | | 04 | AE | 0E B0 00011 | MOVW | #14, CLI_REQ_BLOCK+1 | 1525 |
| | | | 04 | AE | BC D0 00015 | MOVL | #ENABLE_MASK, CLI_REQ_BLOCK+4 | 1526 |
| | | 00000000G | 00 | 5E | DD 0001A | PUSHL | SP | 1531 |
| | | | 02 | 01 | FB 0001C | CALLS | #1, SYSSCLI | |
| | | | | 6C | 91 00023 | CMPB | (AP), #2 | 1537 |
| | | | | 0A | 1F 00026 | BLSSU | 1S | |
| | | | | 08 | AC D5 00028 | TSTL | 8(AP) | |
| | | | | 05 | 13 0002B | BEQL | 1S | |
| | | | 08 | BC | AE D0 0002D | MOVL | CLI_REQ_BLOCK+8, #PREVIOUS_MASK | 1539 |
| | | | 30 | 0C | 50 E8 00032 | BLBS | RETURN_STATUS, 4S | 1544 |
| | | 000388CA | 8F | | 10 ED 00035 | CMPZV | #16, #T2, RETURN_STATUS, #3 | |
| | | | | 29 | 12 0003A | BNEQ | 4S | |
| | | | | 50 | D1 0003C | CMPL | RETURN_STATUS, #231626 | 1550 |
| | | | | 08 | 12 00043 | BNEQ | 2S | |
| | | 50 00000006 | 8F | | D0 00045 | MOVL | #LIB\$INVARG, RETURN_STATUS | |
| | | 00038822 | 8F | | 04 0004C | RET | | |
| | | | | 50 | D1 0004D | CMPL | RETURN_STATUS, #231458 | 1551 |
| | | | | 08 | 12 00054 | BNEQ | 3S | |
| | | | | 50 | 00000006 | MOVL | #LIB\$NOCLI, RETURN_STATUS | |
| | | | | 04 | 0005D | RET | | |
| | | 50 00000006 | 8F | | D0 0005E | MOVL | #LIB\$UNECLIERR, RETURN_STATUS | 1552 |
| | | | | 04 | 00065 | 4S: | | |
| | | | | | | RET | | 1558 |

; Routine Size: 102 bytes, Routine Base: _LIB\$CODE + 0466

1289 1559 1 XSBTTL 'LIBSSBUILD_SYMBOL_NAME - Build a symbol name string'
1290 1560 1 ROUTINE LIBSSBUILD_SYMBOL_NAME (
1291 1561 1 SYMBOL: REF BLOCK [., BYTE],
1292 1562 1 DYNSTR: REF BLOCK [., BYTE],
1293 1563 1 CLI_SYMBOL: REF BLOCK [., BYTE]
1294 1564 1) =
1295 1565 1
1296 1566 1
1297 1567 1
1298 1568 1
1299 1569 1
1300 1570 1
1301 1571 1
1302 1572 1
1303 1573 1
1304 1574 1
1305 1575 1
1306 1576 1
1307 1577 1
1308 1578 1
1309 1579 1
1310 1580 1
1311 1581 1
1312 1582 1
1313 1583 1
1314 1584 1
1315 1585 1
1316 1586 1
1317 1587 1
1318 1588 1
1319 1589 1
1320 1590 1
1321 1591 1
1322 1592 1
1323 1593 1
1324 1594 1
1325 1595 1
1326 1596 1
1327 1597 1
1328 1598 1
1329 1599 1
1330 1600 1
1331 1601 1
1332 1602 1
1333 1603 1
1334 1604 1
1335 1605 1
1336 1606 1
1337 1607 1
1338 1608 1
1339 1609 1
1340 1610 1
1341 1611 1
1342 1612 1
1343 1613 1
1344 1614 1
1345 1615 1

FUNCTIONAL DESCRIPTION:
Using the string pointed to by SYMBOL, LIBSSBUILD_SYMBOL_NAME builds a dynamic string pointed to by CLI_SYMBOL, which has been upcased and has trailing blanks deleted. The string pointed to by SYMBOL is also tested for validity as a symbol name: the first character must be alphabetic or \$ or there must be less than 255 characters in the name (trailing blanks excluded), and there must be at least one non-blank character in the name.
SCRATCH is a string descriptor into which the actual descriptor for the dynamic string will be written. The calling procedure must use this descriptor to deallocate the dynamic string.
If this routine detects an error after it has allocated the dynamic string, it will deallocate the dynamic string thus freeing its caller from that responsibility.

CALLING SEQUENCE:
ret_status.wlc.v = LIBSSBUILD_SYMBOL_NAME (symbol.rt.dx,
dynstr.wt.dx,
cli_symbol.wt.dx)

FORMAL PARAMETERS:
symbol.rt.dx - A string, passed by descriptor, which contains the original symbol name.
dynstr.wt.dx - A string descriptor into which the descriptor for the actual dynamic string used to contain the converted symbol name will be written. The calling procedure must use this descriptor to deallocate the dynamic string.
cli_symbol.wt.dx - A string, passed by descriptor, into which the processed symbol name will be placed. This routine creates a new dynamic string in this descriptor: the contents of the descriptor upon entry are overwritten.

IMPLICIT INPUTS:
LIBSAB_UPCASE the address of an upcase translation table.

IMPLICIT OUTPUTS:
NONE

COMPLETION STATUS: (or ROUTINE VALUE:)

```

1346      1616 1
1347      1617 1
1348      1618 1
1349      1619 1
1350      1620 1
1351      1621 1
1352      1622 1
1353      1623 1
1354      1624 1
1355      1625 1
1356      1626 1
1357      1627 1
1358      1628 1
1359      1629 1
1360      1630 1
1361      1631 1
1362      1632 1
1363      1633 1
1364      1634 2
1365      1635 2
1366      1636 2
1367      1637 2
1368      1638 2
1369      1639 2
1370      1640 2
1371      1641 2
1372      1642 2
1373      1643 2
1374      1644 2
1375      1645 2
1376      1646 2
1377      1647 2
1378      1648 2
1379      1649 2
1380      1650 2
1381      1651 2
1382      1652 2
1383      1653 2
1384      1654 2
1385      1655 2
1386      1656 2
1387      1657 2
1388      1658 2
1389      1659 3
1390      1660 3
1391      1661 4
1392      1662 4
1393      1663 4
1394      1664 4
1395      1665 2
1396      1666 2
1397      1667 2
1398      1668 2
1399      1669 2
1400      1670 2
1401      1671 2
1402      1672 2

      1616 1
      1617 1
      1618 1
      1619 1
      1620 1
      1621 1
      1622 1
      1623 1
      1624 1
      1625 1
      1626 1
      1627 1
      1628 1
      1629 1
      1630 1
      1631 1
      1632 1
      1633 1
      1634 2
      1635 2
      1636 2
      1637 2
      1638 2
      1639 2
      1640 2
      1641 2
      1642 2
      1643 2
      1644 2
      1645 2
      1646 2
      1647 2
      1648 2
      1649 2
      1650 2
      1651 2
      1652 2
      1653 2
      1654 2
      1655 2
      1656 2
      1657 2
      1658 2
      1659 3
      1660 3
      1661 4
      1662 4
      1663 4
      1664 4
      1665 2
      1666 2
      1667 2
      1668 2
      1669 2
      1670 2
      1671 2
      1672 2

      SSS_NORMAL      Normal successful completion
      LIBS_INVARG     Symbol name contains more than 255 characters or is
                      all blanks
      LIBS_INVSYMNAME Symbol name does not begin with a letter
      LIBS_INSIRMEM   Insufficient virtual memory for dynamic string
      Other completion stati from LIBSANALYZE_SDESC.

      SIDE EFFECTS:

      WARNING:
      Although this procedure performs some checks on the validity of
      symbol names passed to it, some symbol name considered valid by
      this procedure will be considered invalid by DCL.

      ---

      BEGIN
      LOCAL
      ADDRESS,           ! a string address
      LENGTH : WORD,    ! a string length
      IN_PTR, OUT_PTR,  ! two pointer variables
      STATUS:           ! a return status value

      + Gather information about input symbol name string
      and build pointer input symbol name string.

      STATUS = LIBSANALYZE_SDESC_R2 (.SYMBOL; LENGTH, ADDRESS );
      IF NOT .STATUS THEN RETURN .STATUS;
      IN_PTR = CHSPTR( .ADDRESS, .LENGTH - 1 );

      + Work backwards from the end of the string to locate
      the first non-blank character; i.e. locate beginning
      of the trailing blanks.

      LENGTH = 1 +
      BEGIN
      DECR J FROM .LENGTH - 1 TO 0 DO
      BEGIN
      IF CHSRCHAR( .IN_PTR ) NEQU XC' ' THEN EXITLOOP .J;
      IN_PTR = CHSPLUST .IN_PTR, -1 );
      END
      END.
      IF .LENGTH EQLU 0 OR .LENGTH GTRU 255 THEN RETURN LIBS_INVSYMNAME;

      + Allocate a dynamic string and setup CLI symbol name
      string descriptor.

      DYNSTR [DSC$B_CLASS] = DSC$K_CLASS_D;

```

```

: 1403      1673  2      DYNSTR [DSC$B_DTYPE] = DSC$K_DTYPE_T;
: 1404      1674  2      DYNSTR [DSC$W_LENGTH] = 0;
: 1405      1675  2      DYNSTR [DSC$A_POINTER] = 0;
: 1406      1676  2      STATUS = LIB$GET1 DD( LENGTH, .DYNSTR );
: 1407      1677  2      IF NOT .STATUS THEN RETURN .STATUS;
: 1408      1678  2      CLI_SYMBOL [DSC$A_POINTER] = .DYNSTR [DSC$A_POINTER];
: 1409      1679  2      CLI_SYMBOL [DSC$W_LENGTH] = .LENGTH;
: 1410      1680  2
: 1411      1681  2      !+ Upcase the symbol name.
: 1412      1682  2      !- IN_PTR = CH$PTR( .ADDRESS );
: 1413      1683  2      OUT_PTR = CH$PTR( .CLI_SYMBOL [DSC$A_POINTER] );
: 1414      1684  2      CH$TRANSLATE( LIB$AB_UPCASE, .LENGTH, .IN_PTR, %C' ', .LENGTH, .OUT_PTR );
: 1415      1685  2
: 1416      1686  2      !+ Check for reasonable first character in symbol name.
: 1417      1687  2      !- IF (CH$RCHAR( .OUT_PTR ) LSSU %C'A' OR CH$RCHAR( .OUT_PTR ) GTRU %C'Z') AND
: 1418      1688  2          (CH$RCHAR( .OUT_PTR ) NEQU %C'$' OR CH$RCHAR( .OUT_PTR ) NEQU %C'_')
: 1419      1689  2          THEN
: 1420      1690  2          BEGIN
: 1421      1691  2              LIB$FREE1 DD( .DYNSTR );
: 1422      1692  2              RETURN LIB$_INVSYMNAME;
: 1423      1693  2          END
: 1424      1694  2
: 1425      1695  3          ELSE
: 1426      1696  3              RETURN SSS_NORMAL;
: 1427      1697  3
: 1428      1698  2
: 1429      1699  2
: 1430      1700  2
: 1431      1701  1      END;

```

! End of routine LIB\$BUILD_SYMBOL_NAME

00FC 00000 LIB\$BUILD_SYMBOL_NAME:

| | | | | | |
|------|-----------|----------------|--------|------------------------|------|
| 5E | 04 | C2 00002 | WORD | Save R2,R3,R4,R5,R6,R7 | 1560 |
| 50 | 04 | AC D0 00005 | SUBL2 | #4, SP | 1649 |
| | 00000000G | 00 16 00009 | MOVL | SYMBOL, R0 | |
| 53 | 52 | D0 0000F | JSB | LIB\$ANALYZE_SDESC_R2 | |
| 6E | 51 | D0 00012 | MOVL | R2, R3 | |
| 41 | 50 | E9 00015 | MOVL | R1, LENGTH | 1650 |
| 52 | 6E | 3C 00018 | BLBC | STATUS, 4\$ | 1651 |
| 52 | 53 | C0 0001B | MOVZWL | LENGTH, R2 | |
| | 52 | D7 0001E | ADDL2 | ADDRESS, R2 | |
| 51 | 6E | 3C 00020 | DECL | IN PTR | |
| | 07 | 11 00023 | MOVZWL | LENGTH, J | 1660 |
| 20 | 62 | 91 00025 | BRB | 2\$ | |
| | 08 | 12 00028 | 1\$: | CMPB (IN_PTR), #32 | 1662 |
| | 52 | D7 0002A | BNEQ | 3\$ | |
| F6 | 51 | F4 0002C | 2\$: | DECL IN_PTR | 1663 |
| 51 | 01 | CE 0002F | SOBGEQ | J-1\$ | 1660 |
| 6E | 51 | 01 A1 00032 | MNEGL | #1, R1 | |
| | 62 | 13 00036 | ADDW3 | #1, R1, LENGTH | 1658 |
| 00FF | 8F | 6E B1 00038 | BEQL | 7\$ | 1666 |
| | 5B | 1A 0003D | CMPW | LENGTH, #255 | |
| | 56 | 08 AC D0 0003F | BGTRU | 7\$ | |
| | | | MOVL | DYNSTR, R6 | 1672 |

| | | | | | | |
|--------------|----|----------------|--------|--|-----------------------|--------|
| 66 020E0000 | 8F | 00 00043 | MOVL | #34471936, (R6) | : 1674 | |
| 04 | A6 | D4 0004A | CLRL | 4(R6) | : 1675 | |
| | 56 | DD 0004D | PUSHL | R6 | : 1676 | |
| | 04 | AE 9F 0004F | PUSHAB | LENGTH | | |
| 00000000G 00 | 00 | 02 FB 00052 | CALLS | #2, LIB\$GET1_DD | | |
| 49 | 50 | E9 00059 | 4\$: | BLBC STATUS, 9\$ | : 1677 | |
| 50 | 0C | AC D0 0005C | MOVL | CLI SYMBOL, R0 | : 1678 | |
| 04 | A0 | 04 A6 D0 00060 | MOVL | 4(R6), 4(R0) | | |
| 60 | 6E | B0 00065 | MOVW | LENGTH, (R0) | : 1679 | |
| 52 | 53 | D0 00068 | MOVL | ADDRESS, IN PTR | : 1684 | |
| 57 | 04 | A0 D0 0006B | MOVL | 4(R0), OUT PTR | : 1685 | |
| 00000000G 00 | 20 | 6E 2E 0006F | MOVTC | LENGTH, (IN PTR), #32, LIB\$AB_UPCASE, - | : 1686 | |
| | 62 | 6E 00078 | | LENGTH, (OUT PTR) | | |
| 41 | 8F | 67 91 0007A | CMPB | (OUT_PTR), #65 | : 1691 | |
| | | 06 1F 0007E | BLSSU | 5\$ | | |
| 5A | 8F | 67 91 00080 | CMPB | (OUT_PTR), #90 | | |
| | | 1C 1B 00084 | BLEQU | 8\$ | | |
| 24 | 67 | 91 00086 | 5\$: | CMPB (OUT_PTR), #36 | : 1692 | |
| | | 06 12 00089 | BNEQ | 6\$ | | |
| 5F | 8F | 67 91 0008B | CMPB | (OUT_PTR), #95 | | |
| | | 11 13 0008F | BEQL | 8\$ | | |
| 00000000G 00 | 00 | 56 DD 00091 | 6\$: | PUSHL | : 1695 | |
| 50 00000000G | 01 | FB 00093 | CALLS | #1, LIB\$FREE1_DD | | |
| | 8F | D0 0009A | 7\$: | MOVL | #LIB\$_INVSYMNAME, R0 | : 1699 |
| | | 04 000A1 | | RET | | |
| 50 | 01 | D0 000A2 | 8\$: | MOVL | #1, R0 | |
| | | 04 000A5 | 9\$: | RET | : 1701 | |

: Routine Size: 166 bytes, Routine Base: _LIB\$CODE + 04CC

LIB\$CLI_CALLBACK LIB\$CLI_CALLBACK - CLI Callback Interface Proce E 11
V04-000 LIB\$BUILD_SYMBOL_NAME - Build a symbol name st 16-Sep-1984 02:22:35 VAX-11 Bliss-32 V4.0-742
LIB\$CLI_CALLBACK LIB\$CLI_CALLBACK - CLI Callback Interface Proce 16-Sep-1984 02:22:35 VAX-11 Bliss-32 V4.0-742
V04-000 LIB\$BUILD_SYMBOL_NAME - Build a symbol name st 14-Sep-1984 13:27:47 [VMSLIB.SRC]LIBCLICAL.B32;1
: 1433 1702 1 END Page 40 (11)
: 1434 1703 0 ELUDOM ! End of module LIB\$CLI_CALLBACK

PSECT SUMMARY

| Name | Bytes | Attributes |
|------------|-------|---|
| _LIB\$CODE | 1394 | NOVEC,NOWRT, RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(2) |

Library Statistics

| File | ----- Symbols ----- | Pages | Processing |
|---------------------------------------|--------------------------------|--------|------------|
| | Total Loaded Percent | Mapped | Time |
| \$_\$255\$DUA28:[SYSLIB]STARLET.L32;1 | 9776 38 0 | 581 | 00:01.0 |

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LISS:LIBCLICAL/OBJ=OBJ\$:LIBCLICAL MSRC\$:LIBCLICAL/UPDATE=(ENH\$:LIBCLICAL
)

: Size: 1394 code + 0 data bytes
: Run Time: 00:34.2
: Elapsed Time: 01:07.4
: Lines/CPU Min: 2990
: Lexemes/CPU-Min: 37106
: Memory Used: 171 pages
: Compilation Complete

0435 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

